

**King Saud University**  
**College of Computer and Information Sciences**  
**Department of Information Technology**

**IT222: Database Principles**



**Jan Burger**

**Phase # 3**

Section #	NAME	ID
View Name: customer		
56385	Raseel Aldawish	443203036
56385	Norah Aljedai	443200841
56385	Doaa Abdul hakim Aldobai	443203882
56385	Ghadah suod alismail	443200501
56385	Raghad Ahmed Rawih Hassan	443204743

**Supervised By:** Abeer Aldrees

## Project Description:

*Jan Burger is an online restaurant that offers its services and promotions through mobile phone applications or websites. The Customers can easily choose the dishes they want to have and add them to the order basket This project aims to develop a database for Jan burger restaurants The purpose of Jan burger DB is to maintain the data that is used & generated to support the online restaurant for the clients*

## View Description:

This view revolves around the customer he/she can order meals, the customer can customize their orders by adding special notes or modifying the ingredients according to their preferences

## Data Requirements:

### Customer:

A customer is the person who orders the meal from the restaurant application . It has a Name, address, Email, Phone Number, password, and identified by Customer ID. Each customer has zero or many orders.

### Orders:

An order is a customer request to buy meals from the restaurant. It has a Date, Status, TotalPrice and identified by Order ID. Each order is owned by one and only one customer.

### Branch:

A location that is a franchise in a restaurant business chain. The branch has an address, contact No. and a unique branch number. Each branch provides multiple item options.

### Item:

An item is the food or drink that can be ordered in the restaurant. It has price, description, extras, calories, type, size, name, and unique item number. Each item is included in zero or many orders and provided by at least one branch.

## **Transaction Requirements:**

### **Data Entry:**

- 1- Customer can enter his name.**
- 2- Customer can enter his address.**
- 3- Customer can enter his phone number.**
- 4- Customer can enter his email.**

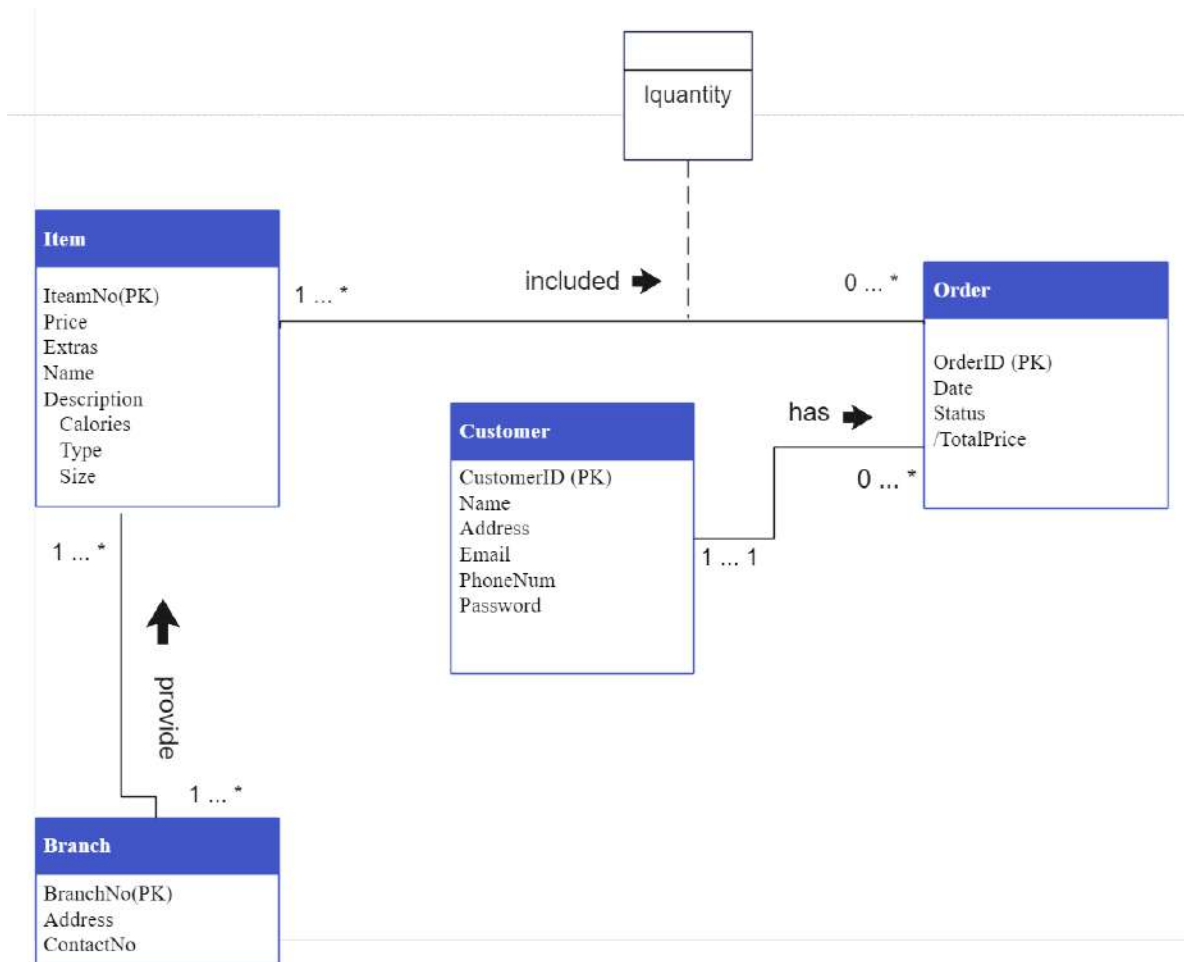
### **Data update/deletion:**

- 1- Customer can update/delete address.**
- 2- Customer can update his account password.**

### **Data Queries:**

- 1. Display customer info.**
- 2. Display all items.**
- 3. Display all orders.**
- 4. Display items from the lowest to highest Price.**
- 5. Display type side dishes items.**
- 6. Display items based on specific size.**
- 7. Display item by specific name.**
- 8. Display orders from the oldest to newest.**
- 9. Display completed status order.**
- 10. Display items from the lowest to highest calorie.**

## Global enhanced entity relationship diagram (EER):



## Relational Schema:

**Item** (ItemNo, Price, Extras, Name, Calories, type, size)

**Primary Key:** ItemNo.

**Branch** (BranchNo, Address, ContactNo)

**Primary Key:** BranchNo.

**Provide** (ItemNo, BranchNo)

**Primary Key:** BranchNo, ItemNo.

**Foreign Key:** BranchNo references Branch (BranchNo).

**Foreign Key:** ItemNo references Item (ItemNo).

**Orders** (OrderID, Date, Status, CustomerID)

**Primary Key:** OrderID.

**Foreign Key:** customerID references Customer (customerID).

Included (OrderID , ItemNo , Quantity)

Primary Key: OrderID, ItemNo.

Foreign Key: OrderID references Orders (OrderID).

Foreign Key: ItemNo references Item (ItemNo).

Customer (customerID, Name, Address, Email, phoneNum, Password)

Primary Key: customerID

#### Data Dictionary showing description of all entities:

Entity Name	Description	Occurrence
Item	Food or drink that can be ordered in the restaurant.	-Each item is included in zero or many orders. -Each item is provided by at least one branch.
Customer	A person who orders the meal from the restaurant application	Each customer has zero or many orders.
Branch	A franchise location of a restaurant business chain.	Each branch provides multiple item options.
Orders	A customer request to buy meals from the restaurant. ☐	-Each order is owned by one and only one customer - Each order includes one or many items.

#### Data Dictionary showing description of all relationships:

Entity Name	Multiplicity	Relationship	Entity Name	Multiplicity
Customer	1..1	has	Orders	0..*
Item	1..*	Included	Orders	0..*
Branch	1..*	provide	Item	1..*

#### Data Dictionary showing description of all attributes:

Entity Name	Attribute	Description	Data Type	Length	Nulls	Multi-Valued	Default Value	Range	PK
Customer	CustomerID	ID uniquely identifies the customer	VARCHAR	10					YES
	Name	The name of the customer	VARCHAR	15	Yes				
	Address	Address of the customer	VARCHAR	60					
	Email	Email of the customer	VARCHAR	30	YES				
	PhoneNum	Phone number of the customer	VARCHAR	10					
	Password	The password of the customer	VARCHAR	20					
Item	ItemNo	Number of the item uniquely identifies the item	VARCHAR	10					YES
	Name	The name of the item	VARCHAR	20					
	Price	The cost of the item	DECIMAL	3					
	Extras	The extras the can be added to the item	VARCHAR	10	YES				
	Description								
	Calories	the calories of item	VARCHAR	4					
	Type	the type of item	VARCHAR	10					
	Size	the size of item	CHARACTER	1			L	S,M,L	

Branch	BranchNo	The number of branches uniquely identifies branch	VARCHAR	10					YES
	Address	The address of the branch	VARCHAR	60					
	ContactNo	The contact number	VARCHAR	10					
Orders	OrderID	The order ID, which uniquely identifies the order	VARCHAR	10					YES
	TotalPrice	The total price of the order	DECIMAL	4					
	Date	The date of the order	DATE						
	Status	The status of the order	VARCHAR	10	YES				

## DB tables creation commands:

```
CREATE TABLE Item (  
    ItemNo VARCHAR(10) not null,  
    Price DECIMAL(3) not null,  
    Extras VARCHAR(10),  
    Name VARCHAR(20) not null,  
    Calories VARCHAR(4) not null,  
    Type VARCHAR(10) not null,  
    Size VARCHAR(1) not null DEFAULT 'L',  
    CHECK (Size IN ('S' , 'M' , 'L')),  
    PRIMARY KEY(ItemNo)  
);
```

```
CREATE TABLE Branch (  
    BranchNo VARCHAR(10) not null ,  
    Address VARCHAR(60) not null,  
    ContactNo VARCHAR(10) not null,  
    PRIMARY KEY(BranchNo)  
);
```

```
CREATE TABLE Customer (  
    customerID VARCHAR(10) not null,  
    Name VARCHAR(15),  
    Address VARCHAR(60) not null,  
    Email VARCHAR(30),  
    phoneNum VARCHAR(10)not null,  
    Password VARCHAR(20)not null,  
    PRIMARY KEY (customerID)  
);
```

```
CREATE TABLE Orders (  
    OrderID VARCHAR(10) not null,  
    TotalPrice DECIMAL (4) not null,  
    Date DATE not null,  
    Status VARCHAR(10),  
    customerID VARCHAR(10) not null,  
    PRIMARY KEY(OrderID),  
    FOREIGN KEY (CustomerID) REFERENCES Customer(customerID)  
);
```

```
CREATE TABLE Included (  
    OrderID VARCHAR(10) not null,  
    ItemNo VARCHAR(10) not null,  
    Quantity INT not null,  
    PRIMARY KEY (OrderID, ItemNo),  
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),  
    FOREIGN KEY (ItemNo) REFERENCES Item(ItemNo)
```



);

```
CREATE TABLE Provide (  
    ItemNo VARCHAR(10) not null,  
    BranchNo VARCHAR(10) not null ,  
    PRIMARY KEY (BranchNo, ItemNo),  
    FOREIGN KEY (BranchNo) REFERENCES Branch(BranchNo),  
    FOREIGN KEY (ItemNo) REFERENCES Item(ItemNo));
```

Schema SQL

```
1 CREATE TABLE Item (  
2     ItemNo VARCHAR(10) not null,  
3     Price DECIMAL(3) not null,  
4     Extras VARCHAR(10),  
5     Name VARCHAR(20) not null,  
6     Calories VARCHAR(4) not null,  
7     Type VARCHAR(10) not null,  
8     Size VARCHAR(1) not null DEFAULT 'L',  
9     CHECK (Size IN ('S', 'M', 'L')),  
10    PRIMARY KEY(ItemNo)  
11 );  
12  
13 CREATE TABLE Branch (  
14     BranchNo VARCHAR(10) not null ,  
15     Address VARCHAR(60) not null,  
16     ContactNo VARCHAR(10) not null,  
17     PRIMARY KEY(BranchNo)  
18 );  
19  
20  
21 CREATE TABLE Customer (  
22     customerID VARCHAR(10) not null,  
23     Name VARCHAR(15),  
24     Address VARCHAR(60) not null,  
25     Email VARCHAR(30),  
26     phoneNum VARCHAR(10) not null,  
27     Password VARCHAR(20) not null,  
28     PRIMARY KEY (customerID)  
29 );  
30  
31 CREATE TABLE Orders (  
32     OrderID VARCHAR(10) not null,  
33     TotalPrice DECIMAL(4) not null,  
34     Date DATE not null,  
35     Status VARCHAR(10),  
36     customerID VARCHAR(10) not null,  
37     PRIMARY KEY (OrderID),  
38     FOREIGN KEY (customerID) REFERENCES Customer(customerID)  
39 );  
40  
41 CREATE TABLE Included (  
42     OrderID VARCHAR(10) not null,  
43     ItemNo VARCHAR(10) not null,  
44     Quantity INT not null,  
45     PRIMARY KEY (OrderID, ItemNo),  
46     FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),  
47     FOREIGN KEY (ItemNo) REFERENCES Item(ItemNo)  
48 );  
49  
50  
51 CREATE TABLE Provide (  
52     ItemNo VARCHAR(10) not null,  
53     BranchNo VARCHAR(10) not null ,  
54     PRIMARY KEY (BranchNo, ItemNo),  
55     FOREIGN KEY (BranchNo) REFERENCES Branch(BranchNo),  
56     FOREIGN KEY (ItemNo) REFERENCES Item(ItemNo);  
57
```

Query successfully executed in 21ms

Text to DDL

DB Fiddle - Crafted with ♥ by Statu200 in the United Kingdom. Terms of Use - Privacy / Cookie Policy - Statu200 Ltd © 2018

Schema SQL

```
11 PASSWORD VARCHAR(20) not null,  
12 PRIMARY KEY (customerID)  
13 };  
14  
15  
16 CREATE TABLE Orders (  
17     OrderID VARCHAR(10) not null,  
18     TotalPrice DECIMAL(4) not null,  
19     Date DATE not null,  
20     Status VARCHAR(10),  
21     customerID VARCHAR(10) not null,  
22     PRIMARY KEY (OrderID),  
23     FOREIGN KEY (customerID) REFERENCES Customer(customerID)  
24 );  
25  
26  
27 CREATE TABLE Included (  
28     OrderID VARCHAR(10) not null,  
29     ItemNo VARCHAR(10) not null,  
30     Quantity INT not null,  
31     PRIMARY KEY (OrderID, ItemNo),  
32     FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),  
33     FOREIGN KEY (ItemNo) REFERENCES Item(ItemNo)  
34 );  
35  
36  
37 CREATE TABLE Provide (  
38     ItemNo VARCHAR(10) not null,  
39     BranchNo VARCHAR(10) not null ,  
40     PRIMARY KEY (BranchNo, ItemNo),  
41     FOREIGN KEY (BranchNo) REFERENCES Branch(BranchNo),  
42     FOREIGN KEY (ItemNo) REFERENCES Item(ItemNo);  
43
```

Text to DDL

DB Fiddle - Crafted with ♥ by Statu200 in the United Kingdom. Terms of Use - Privacy / Cookie Policy - Statu200 Ltd © 2018

### Data insertion commands:

```
INSERT INTO Item VALUES ('365487', 26, 'null', 'Classic beef', '986', 'Meal', 'L');
INSERT INTO Item VALUES ('365478', 34.5, 'Cheese', 'Hotdog', '1170', 'Meal', 'M');
INSERT INTO Item VALUES ('360487', 26, 'Ketchup', 'Kids chicken Meal', '383', 'kids', 'S');
INSERT INTO Item VALUES ('315487', 26, 'null', 'French fries', '487', 'Side', 'L');
```

```
INSERT INTO Branch VALUES ('320', 'Anas Bin M Rd, Alyasmin, Riyadh', '0556392741');
INSERT INTO Branch VALUES ('321', 'Al Thoumamah Rd, Ar Rabi, Riyadh', '0556392755');
INSERT INTO Branch VALUES ('322', 'Uthman Ibn Affan Rd, Al Mughrizat, Riyadh', '0556492755');
INSERT INTO Branch VALUES ('325', 'Prince Meshaal Ibn Abd Al Aziz Rd, Irqah, Riyadh', '0550392735');
```

```
INSERT INTO Provide VALUES ('365487', '320');
INSERT INTO Provide VALUES ('365478', '320');
INSERT INTO Provide VALUES ('360487', '322');
INSERT INTO Provide VALUES ('315487', '325');
```

```
INSERT INTO Customer VALUES ('C001', 'Nasser', '2929 Rayhanah', 'Nasser123@gmail.com', '055672489', 'nasser111');
INSERT INTO Customer VALUES ('C002', 'Ahmad', '2356 Irqah', 'Ahmad20@gmail.com', '055672498', 'ahmad222');
INSERT INTO Customer VALUES ('C003', 'Sara', '1564 Alrayan', 'Sara31@gmail.com', '056672356', 'Sara333');
INSERT INTO Customer VALUES ('C004', 'rawan', '3592 salmeah', 'rawan12@gmail.com', '053445664', 'rawan123');
```

```
INSERT INTO Orders VALUES ('O001', 30, '2023-11-07', 'Completed', 'C001');
INSERT INTO Orders VALUES ('O002', 43, '2023-2-07', 'Completed', 'C002');
INSERT INTO Orders VALUES ('O003', 60, '2023-10-03', 'Preparing', 'C002');
INSERT INTO Orders VALUES ('O004', 30, '2023-6-07', 'Canceled', 'C004');
```

INSERT INTO Included VALUES ( 'O002' , '365487', 3 ) ;  
 INSERT INTO Included VALUES ( 'O003' , '365478', 3 ) ;  
 INSERT INTO Included VALUES ( 'O001' , '360487', 3 ) ;  
 INSERT INTO Included VALUES ( 'O004' , '365487', 3 ) ;

```

Schema SQL *
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
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

```

Database: MySQL v5.7

Run Save Load Example Collaborate

Sign in Have any feedback?

Fiddle Title

60 characters remaining.

Fiddle Description

300 characters remaining.

Private ☐ PRO

This setting cannot be modified after saving the fiddle.

Upgrade to PRO

50% OFF for Early Adopters

Show Keyboard Shortcuts

Schema SQL \*

```

1 CREATE TABLE Item (
2
3   ItemNo VARCHAR(10) not null,
4
5   Price DECIMAL(3) not null,
6
7   Extras VARCHAR(10),
8
9   Name VARCHAR(20) not null,
10
11  Calories VARCHAR(4) not null,
12
13  Type VARCHAR(10) not null,
14
15 )

```

Text to DDL

Query SQL

Query successfully executed in 39ms

1

Results

There are no results to be displayed.

Copy as Markdown

## Data Queries commands and outputs:

### 1) Display customer info.

**SELECT \***  
**FROM Customer ;**

The screenshot shows a SQL IDE interface. On the left, the 'Schema SQL' pane contains the following code:

```
1 CREATE TABLE Item (  
2   ItemNo VARCHAR(10) not null,  
3   Price DECIMAL(3) not null,  
4   Extras VARCHAR(10),  
5   Name VARCHAR(20) not null,  
6   Calories VARCHAR(4) not null,  
7   Type VARCHAR(10) not null,  
8   Size VARCHAR(1) not null DEFAULT 'L',  
9 )
```

On the right, the 'Query SQL' pane shows the executed query:

```
1 SELECT *  
2 FROM Customer ;
```

A green notification box indicates: 'Query successfully executed in 29ms'.

Below the query panes, the 'Results' section shows the output for 'Query #1' with an execution time of 0ms. The results are displayed in a table with 6 columns: customerID, Name, Address, Email, phoneNum, and Password.

customerID	Name	Address	Email	phoneNum	Password
C001	Nasser	2929 Rayhanah	Nasser123@gmail.com	05672489	nasser111
C002	Ahmad	2356 Irqah	Ahmad20@gmail.com	05672498	ahmad222
C003	Sara	1564 Alrayan	Sara31@gmail.com	05672356	Sara333
C004	rawan	3582 salmeah	rawan12@gmail.com	003445064	rawan123

### 2) Display all items.

**SELECT \***  
**FROM Item ;**

The screenshot shows the same SQL IDE interface. The 'Schema SQL' pane contains the same code as in the first screenshot.

The 'Query SQL' pane shows the executed query:

```
1 SELECT *  
2 FROM Item ;
```

A green notification box indicates: 'Query successfully executed in 26ms'.

Below the query panes, the 'Results' section shows the output for 'Query #1' with an execution time of 1ms. The results are displayed in a table with 7 columns: ItemNo, Price, Extras, Name, Calories, Type, and Size.

ItemNo	Price	Extras	Name	Calories	Type	Size
315437	25	null	French fries	487	Side	L
350437	25	Ketchup	Kids chicken Meal	383	kids	S
365478	35	Cheese	Hotdog	1170	meal	M
365437	25	null	Classic beef	986	meal	L

### 3) Display all orders.

**SELECT \***  
**FROM Orders ;**

The screenshot shows a SQL IDE interface. On the left, the 'Schema SQL' pane contains the following DDL:

```
1 CREATE TABLE Item (  
2   ItemNo VARCHAR(10) not null,  
3   Price DECIMAL(5) not null,  
4   Extras VARCHAR(10),  
5   Name VARCHAR(20) not null,  
6   Calories VARCHAR(4) not null,  
7   Type VARCHAR(10) not null,  
8   Size VARCHAR(1) not null DEFAULT 'L',  
9 )  
10
```

On the right, the 'Query SQL' pane contains the query:

```
1 SELECT *  
2 FROM Orders ;  
3
```

A green notification bar at the top right states: 'Query successfully executed in 47ms'.

Below the query panes, the 'Results' section shows the output of the query:

OrderID	TotalPrice	Date	Status	customerID
0001	30	2023-11-07	Completed	C001
0002	43	2023-02-07	Completed	C002
0003	60	2023-10-03	Preparing	C002
0004	30	2023-06-07	Cancelled	C004

### 4) Display items from the lowest to highest Price.

**SELECT \***  
**FROM Item**  
**ORDER BY Price ASC ;**

The screenshot shows the same SQL IDE interface. The 'Schema SQL' pane contains the same DDL as in the previous screenshot.

The 'Query SQL' pane contains the query:

```
1 SELECT *  
2 FROM Item  
3 ORDER BY Price ASC ;
```

A green notification bar at the top right states: 'Query successfully executed in 26ms'.

Below the query panes, the 'Results' section shows the output of the query:

ItemNo	Price	Extras	Name	Calories	Type	Size
315487	26	null	French fries	487	Side	L
360487	26	Ketchup	Kids chicken Meal	383	kids	S
365487	26	null	Classic beef	906	meal	L
365478	35	Cheese	Hottdog	1170	meal	M

## 5) Display type side dishes items.

```
SELECT *
FROM Item
WHERE Type = 'Side';
```

The screenshot shows the DataFiddle interface with the following components:

- Header:** Database: MySQL v5.7, Run, Save, Load Example, Collaborate, Sign in, Have any feedback?
- Left Panel:** Fiddle Title, Fiddle Description, Private Fiddle (PRO), Upgrade to PRO, 50% OFF for Early Adopters, Show Keyboard Shortcuts, Data science interview projects.
- Schema SQL:**

```

115 'Classic beef', '986', 'meal', 'L');
116 INSERT INTO Item VALUES ('365478', 34.5, 'Cheese',
117 'Hotdog', '1170', 'meal', 'M');
118 INSERT INTO Item VALUES ('368487', 26, 'Ketchup',
119 'Kids chicken Meal', '383', 'kids', 'S');
120 INSERT INTO Item VALUES ('315487', 26, 'null',
121 'French fries', '487', 'Side', 'L');
122
123
124

```
- Query SQL:**

```

1 SELECT *
2 FROM Item
3 WHERE Type = 'Side';

```
- Results:**

Query #1 Execution time: 0ms

ItemNo	Price	Extras	Name	Calories	Type	Size
315487	26	null	French fries	487	Side	L

## 6) Display items based on specific size.

```
SELECT *
FROM Item
WHERE Size = 'L';
```

The screenshot shows the DataFiddle interface with the following components:

- Header:** Database: MySQL v5.7, Run, Save, Load Example, Collaborate, Sign in, Have any feedback?
- Left Panel:** Fiddle Title, Fiddle Description, Private Fiddle (PRO), Upgrade to PRO, 50% OFF for Early Adopters, Show Keyboard Shortcuts, Data science practice projects.
- Schema SQL:**

```

8
9 Name VARCHAR(20) not null,
10
11 Calories VARCHAR(4) not null,
12
13 Type VARCHAR(10) not null,
14
15 Size VARCHAR(1) not null DEFAULT 'L',
16
17 CHECK (Size IN ('S', 'M', 'L')),
18
19 PRIMARY KEY(ItemNo)
20

```
- Query SQL:**

```

1 SELECT *
2 FROM Item
3 WHERE Size = 'L';

```
- Results:**

Query #1 Execution time: 1ms

ItemNo	Price	Extras	Name	Calories	Type	Size
315487	26	null	French fries	487	Side	L
365487	26	null	Classic beef	986	meal	L

7) **Display item by specific name.**

```
SELECT *  
FROM Item  
WHERE Name = 'Hotdog' ;
```

The screenshot shows a SQL IDE interface. At the top, there's a blue header bar with buttons for 'Run', 'Save', 'Load Example', and 'Collaborate', along with a 'Sign in' button and a 'Have any feedback?' link. Below the header, the interface is split into two main panels. The left panel, titled 'Schema SQL', displays a table schema for 'Item' with columns: Name (VARCHAR(15)), Address (VARCHAR(20) not null), Email (VARCHAR(30)), phoneNum (VARCHAR(10) not null), Password (VARCHAR(20) not null), and a primary key 'customerID'. The right panel, titled 'Query SQL', shows the query: 'SELECT \* FROM Item WHERE Name = 'Hotdog';'. A green notification box above the query results states 'Query successfully executed in 26ms'. Below the query panels, the 'Results' section shows 'Query #1' with an 'Execution time: 0ms'. The results are displayed in a table with 7 columns: ItemNo, Price, Extras, Name, Calories, Type, and Size. The table contains one row of data for ItemNo 365478.

ItemNo	Price	Extras	Name	Calories	Type	Size
365478	35	Cheese	Hotdog	1170	meal	M

## Work Distribution:

NAME	ID	Percentage	WORK
<i>Raseel Aldawish</i>	<i>443203036</i>	<i>100%</i>	<ul style="list-style-type: none"> <li>-Data Requirements: branch &amp; item.</li> <li>- Data dictionary for entities.</li> <li>-Insertion commands: Item, Branch, and Provide.</li> </ul>
<i>Raghad Ahmed Hassan</i>	<i>443204743</i>	<i>100%</i>	<ul style="list-style-type: none"> <li>-Transaction requirements: Data entry, Data update/deletion, Data queries</li> <li>-Order, included, and customer schemes.</li> </ul>
<i>Doaa Abdul hakim</i>	<i>443203882</i>	<i>100%</i>	<ul style="list-style-type: none"> <li>-Data Requirements: customer order</li> <li>-Data Dictionary for relationships</li> <li>- Data Queries commands and outputs</li> </ul>
<i>Norah Nasser aljedai</i>	<i>443200841</i>	<i>100%</i>	<ul style="list-style-type: none"> <li>-Global enhanced entity relationship diagram</li> <li>-Data Dictionary showing description of all attributes</li> <li>-Insertion commands: Customer ,Order and Included</li> </ul>
<i>Ghadah Suod Alismail</i>	<i>443200501</i>	<i>100%</i>	<ul style="list-style-type: none"> <li>-Project description, view description</li> <li>-Schemes: Item and branch</li> <li>- Data Queries commands and outputs</li> </ul>