

En este sprint, se simula una situación empresarial en la que debes realizar diversas manipulaciones en las tablas de la base de datos. A su vez, tendrás que trabajar con índices y vistas. En esta actividad, continuarás trabajando con la base de datos que contiene información de una empresa dedicada a la venta de productos online. En esta tarea, empezarás a trabajar con información relacionada con tarjetas de crédito.

## Nivel 1:

### Ejercicio 1

Tu tarea es diseñar y crear una tabla llamada "credit\_card" que almacene detalles cruciales sobre las tarjetas de crédito. La nueva tabla debe ser capaz de identificar de forma única cada tarjeta y establecer una relación adecuada con las otras dos tablas ("transaction" y "company"). Después de crear la tabla será necesario que ingreses la información del documento denominado "datos\_introducir\_credit". Recuerda mostrar el diagrama y realizar una breve descripción del mismo.

- Primero creamos la tabla:

```
CREATE TABLE IF NOT EXISTS credit_card(  
  id VARCHAR(20) PRIMARY KEY,  
  iban VARCHAR(50) NULL,  
  pan VARCHAR(30) NULL,  
  pin VARCHAR(4) NULL,  
  cvv VARCHAR(4) NULL,  
  expiring_date VARCHAR(15) NULL);
```

ion Output

Time	Action	Message
18:17:31	CREATE TABLE IF NOT EXISTS credit_card(id VARCHAR(20) PRIMARY KEY, iban VARCHAR(50) NULL, pan VARCHAR(30) NULL, pin VARCHAR(4) NULL, cvv VARCHAR(4) NULL, expiring_date VARCHAR(15) NULL);	0 row(s) affected, 1 warning(s): 1050 Table 'credit_card' already exists

- Le asigné el formato "VARCHAR" a la columna "expiring\_date" en vista de que el formato de los datos proporcionados no coincide con el tipo DATE(YYYY-MM-DD).

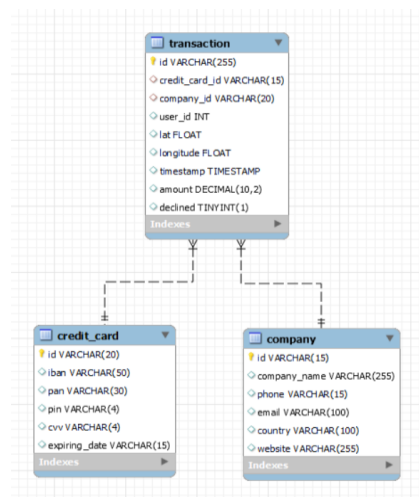
- Cargamos los datos y creé las conexiones entre las tablas:

```
27 • ALTER TABLE transaction  
28   ADD FOREIGN KEY (credit_card_id)  
29   REFERENCES credit_card(id);  
30  
31 -- Ejercicio 2:
```

Output

#	Time	Action	Message
1	18:20:38	ALTER TABLE transaction ADD FOREIGN KEY (credit_card_id) REFERENCES credit_card(id)	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0

- De momento, este es el esquema de nuestra base de datos:



## Ejercicio 2

El departamento de Recursos Humanos ha identificado un error en el número de cuenta del usuario con ID CcU-2938. La información que debe mostrarse para este registro es: R323456312213576817699999. Recuerda mostrar que el cambio se realizó.

- Consultamos los datos:

```
35 • SELECT *
36 FROM credit_card
37 WHERE id = "CcU-2938";
38
```

Result Grid						
Filter Rows: <input type="text"/> Edit:    Export/Import:   Wrap Cell Content:						
id	iban	pan	pin	cvv	expiring_date	
CcU-2938	TR301950312213576817638661	5424465566813633	3257	984	10/30/22	
NULL	NULL	NULL	NULL	NULL	NULL	

credit_card 1 x			
Output			
Action Output			
#	Time	Action	Message
1	10:48:48	SELECT * FROM credit_card WHERE id = "CcU-2938" LIMIT 0, 1000	1 row(s) returned

- Los modificamos y consultamos nuevamente para visualizar los cambios:

```
35 • UPDATE credit_card
36 SET iban = "R323456312213576817699999"
37 WHERE id = "CcU-2938";
38
39 • SELECT *
40 FROM credit_card
41 WHERE id = "CcU-2938";
```

Result Grid						
Filter Rows: <input type="text"/> Edit:    Export/Import:   Wrap Cell Content:						
id	iban	pan	pin	cvv	expiring_date	
CcU-2938	R323456312213576817699999	5424465566813633	3257	984	10/30/22	
NULL	NULL	NULL	NULL	NULL	NULL	

credit_card 2 x			
Output			
Action Output			
#	Time	Action	Message
1	10:53:03	UPDATE credit_card SET iban = "R323456312213576817699999" WHERE id = "CcU-2938"	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0
2	10:53:26	SELECT * FROM credit_card WHERE id = "CcU-2938" LIMIT 0, 1000	1 row(s) returned

## Ejercicio 3

En la tabla "transaction" ingresa un nuevo usuario con la siguiente información:

Id	108B1D1D-5B23-A76C-55EF-C568E49A99DD
credit_card_id	CcU-9999
company_id	b-9999
user_id	9999
lato	829.999
longitud	-117.999
amunt	111.11
declined	0

- El primer paso para crear el nuevo usuario es agregar los id a sus respectivas tablas.
- Y luego si agregamos los valores indicados:

```

46 • INSERT INTO company(id)
47   VALUES('b-9999');
48
49 • INSERT INTO credit_card(id)
50   VALUES('CcU-9999');
51
52 • INSERT INTO transaction(id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
53   VALUES('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', '9999', '829.999', '-117.999', '111.11', '0');
54
55 • SELECT *
56   FROM transaction
57  WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD';

```

Result Grid

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
108B1D1D-5B23-A76C-55EF-C568E49A99DD	CcU-9999	b-9999	9999	829.999	-117.999	NULL	111.11	0

ransaction 5 ×

Output

#	Time	Action	Message
1	11:15:37	INSERT INTO company(id) VALUES(b-9999)	1 row(s) affected
2	11:15:37	INSERT INTO credit_card(id) VALUES(CcU-9999)	1 row(s) affected
3	11:15:37	INSERT INTO transaction(id, credit_card_id, company_id, user_id, lat, longitude, amount, declined) VALU...	1 row(s) affected
4	11:15:37	SELECT * FROM transaction WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD' LIMIT 0, 1000	1 row(s) returned

Se observa que el campo **timestamp** queda en NULL, ya que no se ha proporcionado ningún valor. Entiendo es que esta columna debería capturar automáticamente la fecha y hora de la transacción, información que no tenemos disponible en este caso.

## Ejercicio 4

Desde recursos humanos te solicitan eliminar la columna "pan" de la tabla credit\_card. Recuerda mostrar el cambio realizado.

- Eliminamos la columna y visualizamos la tabla para verificar:

```

62 • ALTER TABLE credit_card
63   DROP COLUMN pan;
64
65 • SELECT*
66   FROM credit_card;

```

Result Grid

id	iban	pin	cvv	expiring_date
CcU-2938	R323456312213576817699999	3257	984	10/30/22
CcU-2945	DO26854763748537475216568689	9080	887	08/24/23
CcU-2952	BG45IVQL52710525608255	4598	438	06/29/21
CcU-2959	CR7242477244335841535	3583	667	02/24/23
CcU-2966	BG72LKTQ15627628377363	4900	130	10/29/24
CcU-2973	PT878N677813502749456146	8760	887	01/30/25

credit\_card 6 ×

Output

#	Time	Action	Message
1	11:19:08	ALTER TABLE credit_card DROP COLUMN pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
2	11:19:34	SELECT* FROM credit_card LIMIT 0, 1000	276 row(s) returned

## Nivel 2

## Ejercicio 1

Elimina de la tabla transacción el registro con ID 02C6201E-D90A-1859-B4EE-88D2986D3B02 de la base de datos.

```
73 • DELETE FROM transaction
74 WHERE ID = '02C6201E-D90A-1859-B4EE-88D2986D3B02';
75
```

Output			
Action Output			
#	Time	Action	Message
1	11:21:54	DELETE FROM transaction WHERE ID = '02C6201E-D90A-1859-B4EE-88D2986D3B02'	1 row(s) affected

- Comprobamos:

```
80 • SELECT *
81 FROM transaction
82 WHERE ID = '02C6201E-D90A-1859-B4EE-88D2986D3B02';
```

Result Grid									
id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined	
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	

Output			
Action Output			
#	Time	Action	Message
1	18:29:10	SELECT * FROM transaction WHERE ID = '02C6201E-D90A-1859-B4EE-88D2986D3B02' LIMIT 0, 1000	0 row(s) returned

## Ejercicio 2

La sección de marketing desea tener acceso a información específica para realizar análisis y estrategias efectivas. Se ha solicitado crear una vista que proporcione detalles clave sobre las compañías y sus transacciones. Será necesaria que crees una vista llamada VistaMarketing que contenga la siguiente información: Nombre de la compañía. Teléfono de contacto. País de residencia. Media de compra realizado por cada compañía. Presenta la vista creada, ordenando los datos de mayor a menor promedio de compra.

- Creamos y visualizamos la vista:

```
93 • CREATE VIEW VistaMarketing AS
94 SELECT c.company_name, c.phone, c.country, AVG(t.amount) AS MediaCompras
95 FROM transaction t JOIN company c
96 ON t.company_id = c.id
97 GROUP BY c.company_name, c.phone, c.country;
98
99 • SELECT *
100 FROM VistaMarketing
101 ORDER BY MediaCompras DESC;
```





Result Grid				
company_name	phone	country	MediaCompras	
Eget Ipsum Ltd	03 67 44 56 72	United States	473.075000	
Non Magna LLC	06 71 73 13 17	United Kingdom	468.345000	
Sed Id Limited	07 28 18 18 13	United States	461.210000	
Justo Eu Arcu Ltd	08 42 56 71 52	Italy	443.635000	
Eget Tincidunt Dui Institute	05 35 93 32 44	Netherlands	442.520000	

Output			
Action Output			
#	Time	Action	Message
1	11:42:54	CREATE VIEW VistaMarketing AS SELECT c.company_name, c.phone, ...	0 row(s) affected
2	11:42:54	SELECT * FROM VistaMarketing ORDER BY MediaCompras DESC LIMIT...	101 row(s) returned


### Ejercicio 3


Filtra la vista VistaMarketing para mostrar sólo las compañías que tienen su país de residencia en "Germany"



```
103 • SELECT *
104 FROM VistaMarketing
105 WHERE country = 'Germany';
```

**Result Grid**   Filter Rows:  | Export:  | Wrap Cell Content: 

	company_name	phone	country	MediaCompras
▶	Ac Fermentum Incorporated	06 85 56 52 33	Germany	206.465000
	Convalis In Incorporated	06 66 57 29 50	Germany	156.730000
	Nunc Interdum Incorporated	05 18 15 48 13	Germany	244.025238
	Augue Foundation	06 88 43 15 63	Germany	240.800000
	Ac Industries	09 34 65 40 60	Germany	289.645000

VistaMarketing 6 

Output 

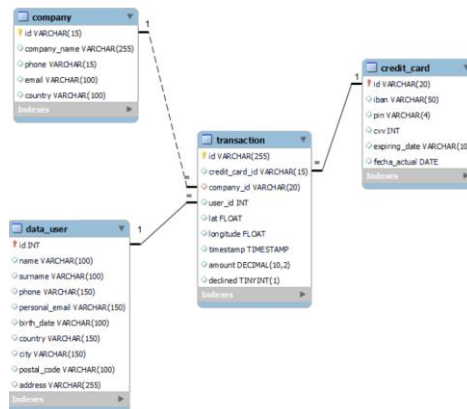
 Action Output 

#	Time	Action	Message
1	11:44:05	SELECT * FROM VistaMarketing WHERE country = 'Germany' LIMIT 0, ...	8 row(s) returned

## Nivel 3

## Ejercicio 1

La próxima semana tendrás una nueva reunión con los gerentes de marketing. Un compañero de tu equipo realizó modificaciones en la base de datos, pero no recuerda cómo las realizó. Te pide que le ayudes a dejar los comandos ejecutados para obtener el siguiente diagrama:



- Primero creamos la tabla “user” y cargamos los datos.

```
106 • CREATE INDEX idx_user_id ON transaction(user_id);
107
108 • CREATE TABLE IF NOT EXISTS user (
109     id INT PRIMARY KEY,
110     name VARCHAR(100),
111     surname VARCHAR(100),
112     phone VARCHAR(150),
113     email VARCHAR(150),
114     birth_date VARCHAR(100),
115     country VARCHAR(150),
116     city VARCHAR(150),
117     postal_code VARCHAR(100),
118     address VARCHAR(255));
119
120 -- Cargamos los datos, dentro de ellos agregué el id '9999' el cual corresponde a uno de los ejercicios anteriores
121 -- Hacemos la conexión entre las tablas
122
123 • ALTER TABLE transaction
124     ADD FOREIGN KEY (user_id)
125     REFERENCES user(id);
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
```

#	Time	Action	Message
1	13:17:58	CREATE INDEX idx_user_id ON transaction(user_id)	0 row(s) affected. Records: 0 Duplicates: 0 Warnings: 0
2	13:17:58	CREATE TABLE IF NOT EXISTS user ( id INT PRIMARY KEY, name VARCHAR(100), surname VARCHAR(100), phone VARCHAR(150), email VARCHAR(150), birth_date VARCHAR(100), country VARCHAR(150), city VARCHAR(150), postal_code VARCHAR(100), address VARCHAR(255));	0 row(s) affected. 1 warning(s): 1050 Table 'user' already exists
3	13:17:58	ALTER TABLE transaction ADD FOREIGN KEY (user_id) REFERENCES user(id);	587 row(s) affected. Records: 587 Duplicates: 0 Warnings: 0

En la carga de datos agregué manualmente la id “9999”, ya que lo habíamos agregado anteriormente a la tabla “transaction”:

```
279 • INSERT INTO user (id) VALUES ('9999');
```

Ahora pasamos a comparar los cambios realizados:

- Con respecto a la tabla company: vemos que se ha eliminado la columna “website”, por lo tanto:

```
130 • ALTER TABLE company
131 DROP COLUMN website;
132
```

Output					
Action Output					
#	Time	Action	Message	Duration / F	
1	13:22:25	ALTER TABLE company DROP COLUMN we...	0 row(s) affected Records: 0 Duplicates: 0 W...	0.031 sec	

- Con respecto a la tabla credit\_card: vemos que el 'cvv' tiene formato INT, que la columna 'expiring\_date' tiene otra cantidad de caracteres y que se crea la columna 'fecha\_actual' con el CURDATE para que siempre nos imprima la fecha de ese momento, por lo tanto, lo ajustamos:

```
142 • ALTER TABLE credit_card
143 MODIFY COLUMN cvv INT,
144 MODIFY COLUMN expiring_date VARCHAR(10),
145 ADD COLUMN fecha_actual DATE DEFAULT(CURDATE());
```

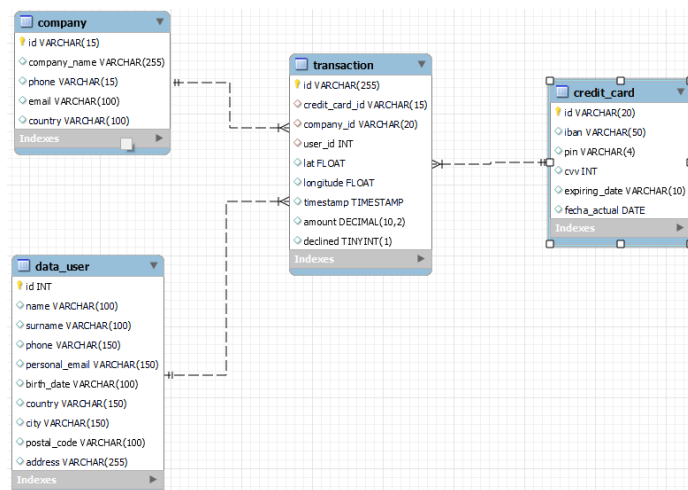
Output					
Action Output					
#	Time	Action	Message	Duration / F	
1	10:44:25	ALTER TABLE credit_card MODIFY COLUMN cvv INT, MODIFY COLUM...	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0		

- Con respecto a la tabla 'user': la tabla tiene un nombre diferente y la columna 'email' tiene el nombre 'personal\_email', por lo tanto:

```
143 • ALTER TABLE user
144 RENAME TO data_user,
145 CHANGE COLUMN email personal_email VARCHAR(150);
...
```

Output					
Action Output					
#	Time	Action	Message	Duration / F	
1	19:22:41	ALTER TABLE user RENAME TO data_user, CHANGE COLUMN email personal_email VARCHAR(150)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0		

Nuestra tabla quedaría de la siguiente manera:



## Ejercicio 2

La empresa también te solicita crear una vista llamada "InformeTecnico" que contenga la siguiente información:

- ID de la transacción
- Nombre del usuario/a
- Apellido del usuario/a
- IBAN de la tarjeta de crédito usada.
- Nombre de la compañía de la transacción realizada.
- Asegúrate de incluir información relevante de ambas tablas y utiliza alias para cambiar de nombre columnas según sea necesario.

Muestra los resultados de la vista, ordena los resultados de forma descendente en función de la variable ID de transacción.

- Creamos la vista y la visualizamos:

```
161 • CREATE VIEW InformeTecnico AS
162 SELECT t.id AS TransactionID, t.timestamp AS TransactionDate, t.amount AS Amount, u.name AS UserName,
163 u.surname AS UserSurname, cc.iban AS IBAN, c.company_name AS Company
164 FROM transaction t
165 JOIN company c
166 ON t.company_id = c.id
167 JOIN data_user u
168 ON t.user_id = u.id
169 JOIN credit_card cc
170 ON t.credit_card_id = cc.id
171 GROUP BY t.id, t.timestamp, t.amount;
172
173 • SELECT *
174 FROM InformeTecnico
175 ORDER BY TransactionID DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [F](#)

TransactionID	TransactionDate	Amount	UserName	UserSurname	IBAN
FE96CE47-8D59-381C-4E18-E3CA3D44E8FF	2021-06-15 00:26:29	480.13	Kenyon	Hartman	DO26854763748537475216568689
FE809ED4-2D86-55AC-C915-929516E4646B	2021-11-09 21:35:40	219.83	Molly	Gilliam	SE2813123487163628531121
FD9CBCCD-8E1E-8DA1-4606-7E3A6F3A5A65	2021-06-13 11:41:17	42.32	Linus	Willis	KW94853327547817578862429556
FD89D51B-AE8D-77DC-E450-B8083FBD3187	2022-03-16 02:35:05	200.72	Hilda	Levy	LT053237077744561475

InformeTecnico 7 x

Output

Action Output

#	Time	Action	Message
1	11:46:07	CREATE VIEW InformeTecnico AS SELECT t.id AS TransactionID, t.time...	0 row(s) affected
2	11:46:07	SELECT * FROM InformeTecnico ORDER BY TransactionID DESC LIM...	586 row(s) returned