

Mailin Adriana Villán Villán  
Fecha entrega: 02/12/2024  
Revisor: Alex Vidal

## Tasca S4.01. Creació de Base de Dades

### NIVELL 1

Descàrrega els arxius CSV, estudia'ls i dissenya una base de dades amb un esquema d'estrella que contingui, almenys 4 taules de les quals puguis realitzar les següents consultes:

#### Creación de la base de datos:

1. Ejecuto el código que me permite crear una nueva base de datos

```
6      -- Creamos la base de datos
7      CREATE DATABASE IF NOT EXISTS transactions;
8      USE transactions;
```

Output

#	Time	Action
1	10:17:02	CREATE DATABASE IF NOT EXISTS transactions

2. Creo la tabla companies

```
11      -- Creo la tabla companies
12      -- drop table companies;
13      CREATE TABLE IF NOT EXISTS companies (
14          company_id VARCHAR(15) PRIMARY KEY NOT NULL,
15          company_name VARCHAR(255),
16          phone VARCHAR(15),
17          email VARCHAR(100),
18          country VARCHAR(100),
19          website VARCHAR(255)
20      );
```

Output

#	Time	Action	Message
1	10:25:25	USE transactions	0 row(s) affected
2	10:25:33	CREATE TABLE IF NOT EXISTS companies ( company_id VARCHAR(15) PRIMARY KEY NOT NULL, company_name VARCHAR(255), phone VAR...	0 row(s) affected

3. Creo la tabla credit\_cards

```

25 • CREATE TABLE IF NOT EXISTS credit_cards (
26     id VARCHAR(15) PRIMARY KEY NOT NULL,
27     user_id INT DEFAULT NULL,
28     iban VARCHAR(50),
29     pan VARCHAR(50),
30     pin VARCHAR(4),
31     cvv int,
32     track1 VARCHAR(255),
33     track2 VARCHAR(255),
34     expiring_date VARCHAR(20)
35 ) ;
36

```

Output

#	Time	Action	Message
✓ 1	10:25:25	USE transactions	0 row(s) affected
✓ 2	10:25:33	CREATE TABLE IF NOT EXISTS companies ( company_id VARCHAR(15) PRIMARY KEY NOT NULL, company_name VARCHAR(255), phone VAR...	0 row(s) affected
✓ 3	10:40:08	CREATE TABLE IF NOT EXISTS credit_cards ( id VARCHAR(15) PRIMARY KEY NOT NULL, user_id INT DEFAULT NULL, iban VARCHAR(50), pa...	0 row(s) affected

4. Creo la tabla users, una sola que incluirá los datos de los 3 CSV (users\_usa, users\_uk, users\_ca) con el objetivo de facilitar las relaciones entre las tablas de dimensión de usuarios y la tabla de hechos transactions.

```

39 • CREATE TABLE IF NOT EXISTS users(
40     id INT PRIMARY KEY,
41     name VARCHAR(100),
42     surname VARCHAR(100),
43     phone VARCHAR(150),
44     email VARCHAR(150),
45     birth_date VARCHAR(100),
46     country VARCHAR(150),
47     city VARCHAR(150),
48     postal_code VARCHAR(100),
49     address VARCHAR(255)
50 );
51

```

Output

#	Time	Action	Message
✓ 1	10:25:25	USE transactions	0 row(s) affected
✓ 2	10:25:33	CREATE TABLE IF NOT EXISTS companies ( company_id VARCHAR(15) PRIMARY KEY NOT NULL, company_name VARCHAR(255), phone VAR...	0 row(s) affected
✓ 3	10:40:08	CREATE TABLE IF NOT EXISTS credit_cards ( id VARCHAR(15) PRIMARY KEY NOT NULL, user_id INT DEFAULT NULL, iban VARCHAR(50), pa...	0 row(s) affected
✓ 4	10:43:35	CREATE TABLE IF NOT EXISTS users( id INT PRIMARY KEY, name VARCHAR(100), surname VARCHAR(100), phone VARCHAR(150), email VA...	0 row(s) affected

## 5. Creo la tabla transactions

```
57 • CREATE TABLE IF NOT EXISTS transactions (  
58     id VARCHAR(255) PRIMARY KEY NOT NULL,  
59     card_id VARCHAR(15) REFERENCES credit_cards(id),  
60     business_id VARCHAR(15) REFERENCES companies(company_id),  
61     timestamp TIMESTAMP,  
62     amount DECIMAL(10, 2),  
63     declined BOOLEAN,  
64     product_ids VARCHAR(15) REFERENCES products(id),  
65     user_id INT REFERENCES users(id),  
66     lat FLOAT,  
67     longitude FLOAT,  
68     FOREIGN KEY (business_id) REFERENCES companies(company_id),  
69     FOREIGN KEY (card_id) REFERENCES credit_cards(id),  
70     FOREIGN KEY (user_id) REFERENCES users(id)  
71 );  
72  
73
```

Output

Action Output

#	Time	Action	Message
1	10:57:44	CREATE TABLE IF NOT EXISTS transactions ( id VARCHAR(255) PRIMARY KEY NOT NULL, card_id VARCHAR(15) REFERENCES credit_cards(id),...	0 row(s) affected

## 6. Al intentar cargar los datos me ha salido lo siguiente:

Error Code: 1290. The MySQL server is running with the --secure-file-priv option so it cannot execute this statement. Para solucionar esto he buscado la ruta de secure-file-priv

```
72 • SHOW VARIABLES LIKE 'secure_file_priv'; -- buscar la ruta de secure-file-priv  
73
```

Result Grid

Variable_name	Value
secure_file_priv	C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\

Result 1 x

Output

Action Output

#	Time	Action
1	11:15:52	SHOW VARIABLES LIKE 'secure_file_priv'

Entonces copio los CSV en la ruta C:\ProgramData\MySQL\MySQL Server 8.0\Uploads

## 7. Cargo los datos de la tabla companies y hago un SELECT \* para mostrar los datos en la tabla

```

78 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\companies.csv'
79 INTO TABLE companies
80 FIELDS TERMINATED BY ','
81 -- ENCLOSED BY '"' -- esto sirve para cuando los valores de los campos estan entre comillas
82 LINES TERMINATED BY '\\n'
83 IGNORE 1 ROWS;
84
85 • SELECT *
86 FROM companies;

```

8. Cargo los datos de la tabla credit\_card y hago un SELECT \* para mostrar los datos en la tabla

```

90 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\credit_cards.csv'
91 INTO TABLE credit_cards
92 FIELDS TERMINATED BY ','
93 -- ENCLOSED BY '"' -- esto sirve para cuando los valores de los campos estan entre comillas
94 LINES TERMINATED BY '\\n'
95 IGNORE 1 ROWS;
96
97 • SELECT *
98 FROM credit_cards;

```

9. Al intentar cargar los datos de la tabla users\_usa me sale el siguiente Error Code: 1262. Row 10 was truncated; it contained more data than there were input columns.

```

102
103 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_usa.csv'
104 INTO TABLE users
105 FIELDS TERMINATED BY ',' ENCLOSED BY '"'
106 LINES TERMINATED BY '\\n'
107 IGNORE 1 ROWS;
108

```

Para solucionar esto he agregado '\r' a la línea de código LINES TERMINATED BY y queda así: '\r\\n'. Posteriormente he ejecutado el código, no me da error pero no me carga los datos

```

103 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_usa.csv'
104 INTO TABLE users
105 FIELDS TERMINATED BY ',' ENCLOSED BY ''
106 LINES TERMINATED BY '\\r\\n'
107 IGNORE 1 ROWS;

```

Output

#	Time	Action	Message
1	12:03:48	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_usa.csv' INTO TABLE users FIELDS TERMINATED BY ',' ENCL...	0 row(s) affected Records: 0 Deleted: 0 Skipped: 0 Warnings: 0

Entonces, he cambiado '\\r\\n' por '\\r\\n' y he conseguido cargar los datos

```

103 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_usa.csv'
104 INTO TABLE users
105 FIELDS TERMINATED BY ',' ENCLOSED BY ''
106 LINES TERMINATED BY '\\r\\n'
107 IGNORE 1 ROWS;
108
109 • SELECT *
110 FROM users;
111

```

Result Grid

	id	name	surname	phone	email	birth_date	country	city	postal_code	address
1	Zeus	Gamble		1-282-581-0551	interdum.enim@protonmail.edu	Nov 17, 1985	United States	Lowell	73544	348-7818 Sagittis St.
2	Garrett	Mcconnell	(718) 257-2412	integer.vitae.nibh@protonmail.org		Aug 23, 1992	United States	Des Moines	59464	903 Sit Ave
3	Ciaran	Harrison	(522) 598-1365	interdum.feugiat@aol.org		Apr 29, 1998	United States	Columbus	56518	736-2063 Tellus St.
4	Howard	Stafford	1-411-740-3269	ornare.egestas@icloud.edu		Feb 18, 1989	United States	Kailua	77417	Ap #545-2244 Erat. Rd.
5	Hayfa	Pierce	1-554-541-2077	et.malesuada.fames@hotmail.org		Sep 26, 1998	United States	Sandy	31564	341-2821 Ultrices Av.

users 4 x

Output

#	Time	Action	Message
1	12:03:48	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_usa.csv' INTO TABLE users FIELDS TERMINATED BY ',' ENCL...	0 row(s) affected Records: 0 Deleted: 0 Skipped: 0 Warnings: 0
2	12:04:58	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_usa.csv' INTO TABLE users FIELDS TERMINATED BY ',' ENCL...	150 row(s) affected Records: 150 Deleted: 0 Skipped: 0 Warnings: 0
3	12:05:36	SELECT * FROM users	150 row(s) returned

10. Cargo los datos de la tabla users\_uk y users\_ca y hago un SELECT \* para mostrar los datos en la tabla

```

112 -- cargo los datos de la tabla users_uk, en user
113
114 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_uk.csv'
115 INTO TABLE users
116 FIELDS TERMINATED BY ',' ENCLOSED BY ''
117 LINES TERMINATED BY '\\r\\n'
118 IGNORE 1 ROWS;
119
120 -- cargo los datos de la tabla users_ca, en user
121
122 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_ca.csv'
123 INTO TABLE users
124 FIELDS TERMINATED BY ',' ENCLOSED BY ''
125 LINES TERMINATED BY '\\r\\n'
126 IGNORE 1 ROWS;
127
128 • SELECT *
129 FROM users;
130

```

Result Grid

Filter Rows:

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

1001

1002

1003

1004

1005

1006

1007

1008

1009

1010

1011

1012

1013

1014

1015

1016

1017

1018

1019

1020

1021

1022

1023

1024

1025

1026

1027

1028

1029

1030

1031

1032

1033

1034

1035

1036

1037

1038

1039

1040

1041

1042

1043

1044

1045

1046

1047

1048

1049

1050

1051

1052

1053

1054

1055

1056

1057

1058

1059

1060

1061

1062

1063

1064

1065

1066

1067

1068

1069

1070

1071

1072

1073

1074

1075

1076

1077

1078

1079

1080

1081

1082

1083

1084

1085

1086

1087

1088

1089

1090

1091

1092

1093

1094

1095

1096

1097

1098

1099

1100

1101

1102

1103

1104

1105

1106

1107

1108

1109

1110

1111

1112

1113

1114

1115

1116

1117

1118

1119

1120

1121

1122

1123

1124

1125

1126

1127

1128

1129

1130

1131

1132

1133

1134

1135

1136

1137

1138

1139

1140

1141

1142

1143

1144

1145

1146

1147

1148

1149

1150

1151

1152

1153

1154

1155

1156

1157

1158

1159

1160

1161

1162

1163

1164

1165

1166

1167

1168

1169

1170

1171

1172

1173

1174

1175

1176

1177

1178

1179

1180

1181

1182

1183

1184

1185

1186

1187

1188

1189

1190

1191

1192

1193

1194

1195

1196

1197

1198

1199

1200

1201

1202

1203

1204

1205

1206

1207

1208

1209

1210

1211

1212

1213

1214

1215

1216

1217

1218

1219

1220

1221

1222

1223

1224

1225

1226

1227

1228

1229

1230

1231

1232

1233

1234

1235

1236

1237

1238

1239

1240

1241

1242

1243

1244

1245

1246

1247

1248

1249

1250

1251

1252

1253

1254

1255

1256

1257

1258

1259

1260

1261

1262

1263

1264

1265

1266

1267

1268

1269

1270

1271

1272

1273

1274

1275

1276

1277

1278

1279

1280

1281

1282

1283

1284

1285

1286

1287

1288

1289

1290

1291

1292

1293

1294

1295

1296

1297

1298

1299

1300

1301

1302

1303

1304

1305

1306

1307

1308

1309

1310

1311

1312

1313

1314

1315

1316

1317

1318

1319

1320

1321

1322

1323

1324

1325

1326

1327

1328

1329

1330

1331

1332

1333

1334

1335

1336

1337

1338

1339

1340

1341

1342

1343

1344

1345

1346

1347

1348

1349

1350

1351

1352

1353

1354

1355

1356

1357

1358

1359

1360

1361

1362

1363

1364

1365

1366

1367

1368

1369

1370

1371

1372

1373

1374

1375

1376

1377

1378

1379

1380

1381

1382

1383

1384

1385

1386

1387

1388

1389

1390

1391

1392

1393

1394

1395

1396

1397

1398

1399

1400

1401

1402

1403

1404

1405

1406

1407

1408

1409

1410

1411

1412

1413

1414

1415

1416

1417

1418

1419

1420

1421

1422

1423

1424

1425

1426

1427

1428

1429

1430

1431

1432

1433

1434

1435

1436

1437

1438

1439

1440

1441

1442

1443

1444

1445

1446

1447

1448

1449

1450

1451

1452

1453

1454

1455

1456

1457

1458

1459

1460

1461

1462

1463

1464

1465

1466

1467

1468

1469

1470

1471

1472

1473

1474

1475

1476

1477

1478

1479

1480

1481

1482

1483

1484

1485

1486

1487

1488

1489

1490

1491

1492

1493

1494

1495

1496

1497</

11. Al intentar cargar los datos de la tabla transactions me sale el siguiente Error Code:

```

136 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\transactions.csv'
137 INTO TABLE transactions
138 FIELDS TERMINATED BY ','
139 ENCLOSED BY ''
140 LINES TERMINATED BY '\\r\\n'
141 IGNORE 1 ROWS;

```

#	Time	Action	Message
1	12:26:52	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\transactions.csv' INTO TABLE transactions FIELDS TERMINATED BY '...' Error Code: 1261. Row 1 doesn't contain data for all columns	Error Code: 1261. Row 1 doesn't contain data for all columns

Para corregirlo he cambiado en la línea de código FILES TERMINATED BY la ',' por ';' porque al visualizar los datos del CSV en Notepad los valores de los campos están separados por punto y coma y no por coma.

Entonces, se cargan los datos en la tabla transactions y hago un SELECT \* para mostrarlos.

```

136 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\transactions.csv'
137 INTO TABLE transactions
138 FIELDS TERMINATED BY ';'
139 ENCLOSED BY '"'
140 LINES TERMINATED BY '\\r\\n'
141 IGNORE 1 ROWS;
142
143 • SELECT *
144 FROM transactions;

```

id	card_id	business_id	timestamp	amount	declined	product_ids	user_id	lat	longitude
02C6201E-D90A-1859-84EE-88D2986D3802	CdU-2938	b-2362	2021-08-28 23:42:24	466.92	0	71, 1, 19	92	81.9185	-12.5276
0466A42E-47CF-8D24-FD01-C0B689713128	CdU-4219	b-2302	2021-07-26 07:29:18	49.53	0	47, 97, 43	170	-43.9695	-117.525
063FBA79-99EC-66FB-29F7-25726D1764A5	CdU-2987	b-2250	2022-01-06 21:25:27	92.61	0	47, 67, 31, 5	275	-81.2227	-129.05
0668296C-CDB9-A883-76BC-2E4C4F8C8AE	CdU-3743	b-2618	2022-01-26 02:07:14	394.18	0	89, 83, 79	265	-34.3593	-100.596
06CD9AA5-9B42-D684-DDDD-A5E394FEB499	CdU-2959	b-2346	2021-10-26 23:00:01	279.93	0	43, 31	92	33.7381	158.298

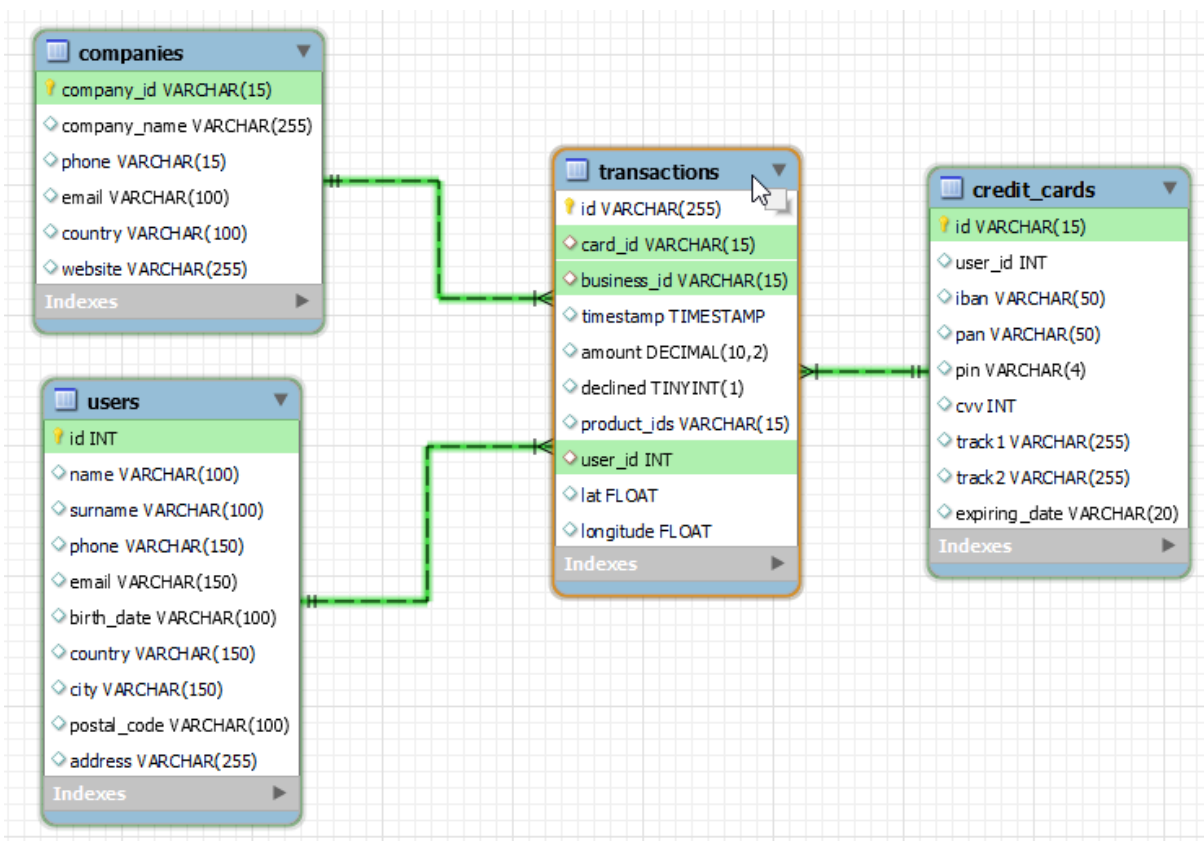
transactions 6 x

Output

Action Output

#	Time	Action	Message
1	12:34:16	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\transactions.csv' INTO TABLE transactions FIELDS TERMINATED BY ...	587 row(s) affected Records: 587 Deleted: 0 Skipped: 0 Warnings: 0
2	12:35:35	SELECT * FROM transactions	587 row(s) returned

12. Creo el modelo de la base de datos: como se puede observar la BD tiene un **modelo en estrella** donde la tabla de hechos es la tabla transacciones y las tablas de dimensión son: companies, users y credit\_cards.



La tabla companies: tiene 100 registros y los siguientes 6 campos:

- company\_id: es la PK
- company\_name: es el nombre de la compañía
- phone: es el teléfono de la compañía
- email: es la dirección de correo electrónico
- country: el país donde se encuentra la compañía
- website: la página web de la compañía

La tabla users: tiene 275 registros y los siguientes 10 campos:

- id: es la PK
- name: es el nombre del usuario
- surname: es el apellido del usuario
- phone: es el número de teléfono del usuario
- email: es la dirección de correo electrónico
- birth\_date: es la fecha de nacimiento
- country: el país del usuario
- city: es la ciudad
- postal\_code: es el código postal del usuario
- address: es la dirección del usuario

La tabla credit\_cards: tiene 275 registros y los siguientes 9 campos:

- id: es la **PK**. Podría llamarse id\_credit\_card para evitar confusiones con el campo id de las tablas transactions y users
- user\_id: es el id del usuario
- iban: es el International Bank Account Number
- pan: es el 'Personal Account Number' o número de la tarjeta de crédito
- pin: es el número secreto de 4 dígitos
- cvv: es el Valor de Verificación de Tarjeta
- track1
- track2
- expiring\_date: es la fecha de caducidad de la tarjeta

La tabla transactions: tiene 587 registros y los siguientes 10 campos:

- id: es la **PK**. Podría llamarse id\_transactions para evitar confusiones con el campo id de las tablas credit\_cards y users
- card\_id: **es FK** y el campo por el cual se relaciona la tabla transactions con la tabla credit\_cards, es el identificador de la tarjeta de crédito con la que se realiza cada transacción
- business\_id: **es FK** y el campo por el cual se relaciona la tabla transactions con la tabla companies, es el identificador de la compañía.
- timestamp: es la fecha y la hora en la que se realiza la transacción
- amount: cantidad de la transacción
- declined: indica si la transacción ha sido rechazada (1) o aceptada (0)
- products\_id: es el identificador del producto adquirido en cada transacción
- user\_id: **es FK** y el identificador del usuario que realiza cada transacción
- lat: es la latitud desde donde se ha realizado la transacción
- longitude: es la longitud desde donde se ha realizado la transacción

## EXERCICI 1

Realitza una subconsulta que mostri tots els usuaris amb més de 30 transaccions utilitzant almenys 2 taules.



En este ejercicio, en la subconsulta, he hecho un SELECT de la tabla transactions, he agrupado por el id de usuario y he filtrado (HAVING) los usuarios que tienen más de 30 transacciones, utilizando la función de agregación COUNT para contar la cantidad de transacciones.

```

147 • SELECT name, surname, email
148 FROM users
149 WHERE id IN (SELECT user_id
150             FROM transactions
151             GROUP BY user_id
152             HAVING COUNT(id)>30)
153 ORDER BY name;

```

name	surname	email
Hedwig	Gilbert	sem.eget@icloud.edu
Kenyon	Hartman	convallis.ante.lectus@yahoo.com
Lynn	Riddle	vitae.aliquet@outlook.edu
Ocean	Nelson	aenean@yahoo.com

#	Time	Action	Message
1	12:07:52	SELECT name, surname, email FROM users WHERE id IN (SELECT user_id FROM transactions GROUP BY user_id HAVING COUNT(id)>30) ORDER BY name;	4 row(s) returned

## EXERCICI 2

Mostra la mitjana d'amount per IBAN de les targetes de crèdit a la companyia Donec Ltd, utilitza almenys 2 taules.

He utilizado la función de agregación AVG para calcular la media de la cantidad de las transacciones y he agrupado por iban. He hecho JOIN entre las tablas transactions, credit\_cards y companies. He filtrado por el nombre de la compañía. También he usado la función ROUND para redondear el resultado de la cantidad media.

```

162 • SELECT company_name, iban, ROUND(AVG(amount),2)
163 FROM transactions
164 JOIN credit_cards ON card_id = credit_cards.id
165 JOIN companies ON business_id = companies.company_id
166 WHERE company_name = 'Donec Ltd'
167 GROUP BY iban;
168

```

company_name	iban	ROUND(AVG(amount),2)
Donec Ltd	PT87806228135092429456346	203.72

#	Time	Action	Message
1	07:42:28	SELECT company_name, iban, ROUND(AVG(amount),2) FROM transactions JOIN credit_cards ON card_id = credit_cards.id JOIN companies ON business_id = companies.company_id WHERE company_name = 'Donec Ltd' GROUP BY iban;	1 row(s) returned

## NIVELL 2

Crea una nova taula que reflecteixi l'estat de les targetes de crèdit basat en si les últimes tres transaccions van ser declinades i genera la següent consulta:

Para resolver este ejercicio lo primero que hice fue crear la tabla estat\_targetes

```
177 CREATE TABLE IF NOT EXISTS estat_targetes (  
178     card_id VARCHAR(15) PRIMARY KEY NOT NULL,  
179     estado VARCHAR(25)  
180 );
```

Output

Action Output

#	Time	Action	Message
1	07:48:30	CREATE TABLE IF NOT EXISTS estat_targetes ( card_id VARCHAR(15) PRIMARY KEY N...	0 row(s) affected

Para cargar los datos dentro de la tabla estat\_targetes he utilizado un CTE (Common Table Expression) definido con WITH, que puedo usar en la consulta principal o en otras subconsultas de la misma consulta. Es una tabla temporal que sólo existe durante la ejecución de la consulta y no se guarda permanentemente en la base de datos. Es una tabla intermedia.

Utilizo la función ROW\_NUMBER para asignar un número de fila a cada transacción de cada tarjeta empezando desde 1. La función PARTITION BY reinicia la asignación del número de fila dentro de cada partición (cada tarjeta) de forma independiente. Cada partición se ordena (ORDER BY) por la fecha de la transacción de mayor a menor (DESC).

Filtro (WHERE) las 3 transacciones más recientes para cada tarjeta y posteriormente utilizo CASE para definir el estado de la tarjeta. Cuando se tienen las últimas 3 transacciones declinadas(SUM), la tarjeta está inactiva, de lo contrario estará activa.

```

183 • INSERT INTO estat_targetes (card_id, estado)
184 WITH transacciones_tarjetas AS (
185     SELECT card_id, declined,
186           ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS row_transacciones
187     FROM transactions
188 )
189 SELECT card_id,
190        CASE
191            WHEN SUM(declined) = 3 THEN 't_inactiva'
192            ELSE 't_activa'
193        END AS estado_tarjeta
194 FROM transacciones_tarjetas
195 WHERE row_transacciones <= 3
196 GROUP BY card_id;

```

Output

#	Time	Action	Message
✓ 1	08:10:42	CREATE TABLE IF NOT EXISTS estat_targetes ( card_id VARCHAR(15) PRIMARY KEY N...	0 row(s) affected
✓ 2	08:10:52	INSERT INTO estat_targetes (card_id, estado) WITH transacciones_tarjetas AS ( SELEC...	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0

Muestro los datos cargados en la tabla creada recientemente:

```

198 • SELECT *
199 FROM estat_targetes;

```

Result Grid

card_id	estado
CdU-2938	t_activa
CdU-2945	t_activa
CdU-2952	t_activa
CdU-2959	t_activa
CdU-2966	t_activa

estat\_targetes 54 x

Output

#	Time	Action	Message
✓ 2	08:10:52	INSERT INTO estat_targetes (card_id, estado) WITH transacciones_tarjetas AS ( SEL...	275 row(s) affected F
✓ 3	08:11:48	SELECT * FROM estat_targetes	275 row(s) returned

Creo la relación entre la tabla estat\_targetes y credit\_cards:

```

202 • ALTER TABLE estat_targetes
203     ADD FOREIGN KEY (card_id) REFERENCES credit_cards(id);

```

Output

#	Time	Action	Message
✓ 1	09:02:57	ALTER TABLE estat_targetes ADD FOREIGN KEY (card_id) REFERENCES credit_cards(d)	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0

## EXERCICI 1

Quantes targetes estan actives?

```
208 • SELECT COUNT(estado) AS tarjetas_activas
209 FROM estat_targetes
210 WHERE estado = 't_activa';
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
tarjetas_activas			
275			

Result 57 ×

Output

Action Output

#	Time	Action	Message
✓ 1	09:08:21	SELECT COUNT(estado) AS tarjetas_activas FROM estat_targetes WHERE estado = t_acti...	1 row(s) returned

## NIVELL 3

Crea una taula amb la qual puguem unir les dades del nou arxiu products.csv amb la base de dades creada, tenint en compte que des de transaction tens product\_ids. Genera la següent consulta:

### 1. Creo la tabla products

```
215 • CREATE TABLE IF NOT EXISTS products (
216     id VARCHAR(15) PRIMARY KEY NOT NULL,
217     product_name VARCHAR(150),
218     price VARCHAR(100),
219     colour VARCHAR(100),
220     weight FLOAT,
221     warehouse_id VARCHAR(10)
222 );
```

Output

Action Output

#	Time	Action	Message
✓ 1	00:25:35	CREATE TABLE IF NOT EXISTS products ( id VARCHAR(15) PRIMARY KEY NOT NULL, ...	0 row(s) affected

## 2. Cargo los datos en la tabla products

```
226 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\products.csv'
227 INTO TABLE products
228 FIELDS TERMINATED BY ','
229 -- ENCLOSED BY '"' -- esto sirve para cuando los valores de los campos estan entre comillas
230 LINES TERMINATED BY '\\n'
231 IGNORE 1 ROWS;
```

Output

#	Time	Action	Message
1	00:25:35	CREATE TABLE IF NOT EXISTS products ( id VARCHAR(15) PRIMARY KEY NOT NULL, ...	0 row(s) affected
2	00:26:32	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\products.cs...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0

Tabla products: tiene 100 registros y los siguientes 6 campos:

- id: es la PK
- product\_name: es el nombre del producto
- price: es el precio del producto
- colour: color del producto
- weight: peso del producto
- warehouse\_id: identificador warehouse del producto

## 3. Creo la tabla transactions\_products

Debido a que el campo products\_ids de la tabla transactions, en algunos casos, contiene más de 1 identificador del producto, es necesario separar los products\_ids para poder realizar la relación con la tabla products. Por este motivo he creado la tabla transactions\_products.

```
238 -- Creo la tabla transactions_productos
239 • CREATE TABLE IF NOT EXISTS transactions_products (
240     id MEDIUMINT NOT NULL AUTO_INCREMENT PRIMARY KEY,
241     transaction_id VARCHAR(255),
242     product_id VARCHAR(15),
243     FOREIGN KEY (transaction_id) REFERENCES transactions(id),
244     FOREIGN KEY (product_id) REFERENCES products(id)
245 );
```

Output

#	Time	Action	Message
2	00:26:32	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\products.c...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0
3	00:27:48	CREATE TABLE IF NOT EXISTS transactions_products ( id MEDIUMINT NOT NULL AUT...	0 row(s) affected

## 4. Cargo los datos en la tabla transactions\_products. Este punto lo he hecho de 2 maneras. La solución 1 (previa a la corrección) y la solución 2 (posterior a la corrección)

### Solución 1

Selecciono el id de cada transacción y de cada producto de la tabla transactions. He utilizado la función CAST para convertir el id de cada producto en un número entero positivo.

Uso JSON\_TABLE para transformar la cadena de products\_ids, separados por coma, en un formato que se puede usar como tabla temporal. Primero se convierte la cadena en formato JSON con REPLACE y CONCAT para poner comillas a cada id para posteriormente descomponer la lista de ids con JSON\_TABLE.

```
247 • INSERT INTO transactions_products (transaction_id, product_id)
248     SELECT id,
249            CAST(jt.product_id AS UNSIGNED) AS Product_id
250     FROM transactions
251     JOIN
252     JSON_TABLE(
253         CONCAT('["', REPLACE(product_ids, ',', '","'), "']'),
254         '$[*]' COLUMNS (
255             product_id VARCHAR(15) PATH '$'
256         )
257     ) AS jt;
```

Output

#	Time	Action	Message
3	00:27:48	CREATE TABLE IF NOT EXISTS transactions_products ( id MEDIUMINT NOT NULL AUT...	0 row(s) affected
4	00:28:50	INSERT INTO transactions_products (transaction_id, product_id) SELECT id, CAST(jt.prod...	1457 row(s) affected Records: 1457 Duplicates: 0 Warnings: 0

Tabla transactions\_products: tiene 1457 registros y los siguientes 3 campos:

- id: es la PK
- transaction\_id: identificador de cada transacción
- product\_id: identificador de cada producto

5. Hago un SELECT de la tabla previamente creada para ver su contenido

```
259 • SELECT *
260     FROM transactions_products;
```

Result Grid

	id	transaction_id	product_id
1	02C6201E-D90A-1859-84EE-88D2986D3B02	71	
2	02C6201E-D90A-1859-84EE-88D2986D3B02	1	
3	02C6201E-D90A-1859-84EE-88D2986D3B02	19	
4	0466A42E-47CF-8D24-FD01-C0B689713128	47	
5	0466A42E-47CF-8D24-FD01-C0B689713128	97	

transactions\_products 64 x

Output

#	Time	Action	Message
4	00:28:50	INSERT INTO transactions_products (transaction_id, product_id) SELECT id, CAST(jt.pro...	1457 row(s) affected Records: 1457 Duplicates: 0 Warnings: 0
5	00:29:33	SELECT * FROM transactions_products	1457 row(s) returned

## Solución 2

Para saber cuántos productos se han vendido por transacción, contaré cuántas comas hay en cada cadena de caracteres. El número de registros será igual al número de comas +1. El cálculo lo he realizado contando la longitud de cada registro en product\_ids y le resto la longitud de ese mismo registro sustrayendo los caracteres ','. El resultado de éste cálculo lo almaceno en la tabla temporal transation\_num\_product.

```
263 • CREATE TABLE transation_num_product (
264     id VARCHAR(255),
265     product_ids VARCHAR(255),
266     num_product INT
267 )
268 SELECT id, product_ids,
269     LENGTH(product_ids) - LENGTH(REPLACE(product_ids, ',')) + 1 AS num_product
270 FROM transactions;
271
272 • SELECT *
273 FROM transation_num_product;
```

Result Grid

id	product_ids	num_product
02C6201E-D90A-1859-B4EE-88D2986D3B02	71, 1, 19	3
0466A42E-47CF-8D24-FD01-C0B689713128	47, 97, 43	3
063FBA79-99EC-66FB-29F7-25726D1764A5	47, 67, 31, 5	4
0668296C-CD89-A883-76BC-2E4C44F8C8AE	89, 83, 79	3
06CD9AA5-9B42-D684-DDDD-A5E394FEB999	43, 31	2
07A46D48-31A3-7E87-65B9-0DA902AD109F	47, 23	2
09DE92CE-6F27-2B87-13B5-9385B2B388E2	67, 7	2

transation\_num\_product 10 x

Output

Action Output

#	Time	Action	Message
1	13:14:54	CREATE TABLE transation_num_product ( id VARCHAR(255), product_ids VARCHAR(255), num_product INT ) SELECT id, product_ids, LENGTH...	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0
2	13:14:59	SELECT * FROM transation_num_product	587 row(s) returned

Creo otra tabla temporal num\_control que contendrá la cantidad de productos que se venden por transacción

```
278 • CREATE TEMPORARY TABLE IF NOT EXISTS num_control
279     SELECT DISTINCT(num_product) num
280     FROM transation_num_product
281     ORDER BY num ASC;
282
283 -- Muestro los datos de la nueva tabla de control
284 • SELECT *
285 FROM num_control;
```

Result Grid

num
1
2
3
4

num\_control 11 x

Output

Action Output

#	Time	Action	Message
1	13:38:36	SELECT * FROM num_control	4 row(s) returned

La extracción de cada identificador de producto del campo product\_ids de la tabla transactions lo he hecho con la función SUBSTRING\_INDEX hasta la primera ocurrencia del signo de separación (coma).

SUBSTRING\_INDEX(product\_ids,',',1)

Después de introducir en transactions\_products el primer o único producto del campo product\_ids de cada transacción, se prepara una segunda consulta en la que se aumenta el parámetro de ocurrencia del separador para ampliar así la cadena de texto hasta el segundo producto y, a esa cadena, se le aplica otro SUBSTRING\_INDEX() para extraer el id de producto que queda a la derecha de la cadena de texto. Esto se consigue cambiando el parámetro de ocurrencia a -1.

Seguidamente, se selecciona el segundo producto de products\_id para los registros que tienen un num\_product >1. Esto se consigue al aplicar la función hasta la segunda coma y, sobre esta cadena (que contiene los productos hasta la segunda coma), se le aplica de nuevo la función, pero recortando desde la derecha hasta la primera coma que se encuentra, es decir:

SUBSTRING\_INDEX(SUBSTRING\_INDEX(product\_ids,',',2),',',-1)

Posteriormente, el tercer producto de products\_id para los registros con num\_product >2, que, al aplicar el mismo procedimiento, se obtiene de la siguiente manera:

SUBSTRING\_INDEX(SUBSTRING\_INDEX(product\_ids,',',3),',',-1)

Se sigue el mismo procedimiento hasta llegar al número máximo de productos; es decir, en nuestro caso, hasta num\_control.num = 4.

```
289 • CREATE TABLE IF NOT EXISTS transactions_products (  
290     transaction_id VARCHAR(255),  
291     product_id VARCHAR(15),  
292     FOREIGN KEY (transaction_id) REFERENCES transactions(id),  
293     FOREIGN KEY (product_id) REFERENCES products(id)  
294 );  
295  
296 -- cargo los datos en la tabla transactions_products  
297 • INSERT INTO transactions_products  
298     SELECT id, TRIM(SUBSTRING_INDEX(SUBSTRING_INDEX(product_ids,',',num),',',-1))  
299     FROM transacion_num_product  
300     JOIN num_control ON num_product >= num;  
301  
302 • SELECT *  
303     FROM transactions_products;
```

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

Fetch rows:

transaction_id	product_id
02C6201E-D90A-1859-B4EE-88D2986D3802	19
02C6201E-D90A-1859-B4EE-88D2986D3802	1
02C6201E-D90A-1859-B4EE-88D2986D3802	71
0466A42E-47CF-8D24-FD01-C0B689713128	43
0466A42E-47CF-8D24-FD01-C0B689713128	97
0466A42E-47CF-8D24-FD01-C0B689713128	47
063FBA79-99EC-66FB-29F7-25726D1764A5	5

transactions\_products 12 x

Output

Action Output

#	Time	Action	Message
1	13:52:38	CREATE TABLE IF NOT EXISTS transactions_products ( transaction_id VARCHAR(255), product_id VARCHAR(15), FOREIGN KEY (transaction_id) ...	0 row(s) affected
2	13:52:44	INSERT INTO transactions_products SELECT id, TRIM(SUBSTRING_INDEX(SUBSTRING_INDEX(product_ids,',',num),',',-1)) FROM transacion_num_pr...	1457 row(s) affected Records: 1457 Duplicates: 0 Warnings: 0
3	13:53:34	SELECT * FROM transactions_products	1457 row(s) returned

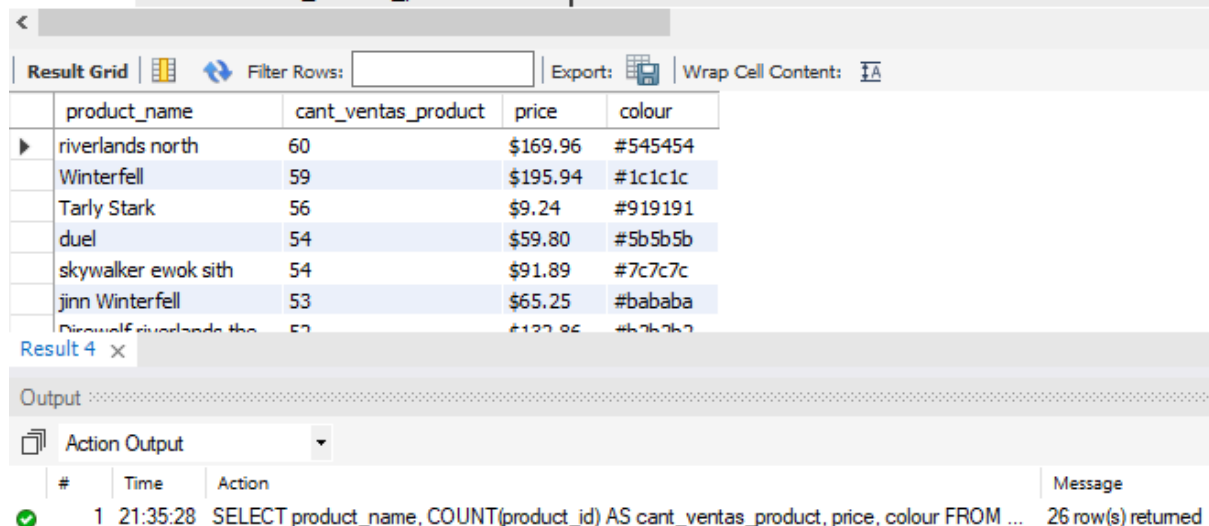


## EXERCICI 1

Necessitem conèixer el nombre de vegades que s'ha venut cada producte.

En este ejercicio he utilizado la función de agregación COUNT (del campo product.id de la tabla transactions\_products) para saber el número de veces que se ha vendido un producto y lo agrupo por el identificador del producto (tabla products). He filtrado (WHERE) por las transacciones aceptadas. También muestro en el resultado de la consulta, el color y el precio del producto, ya que existen varios productos con el mismo nombre pero color y precio diferentes.

```
309 • SELECT product_name, COUNT(product_id) AS cant_ventas_product, price, colour
310 FROM products
311 JOIN transactions_products ON products.id = transactions_products.product_id
312 JOIN transactions ON transactions.id = transactions_products.transaction_id
313 WHERE declined = 0
314 GROUP BY products.id
315 ORDER BY cant_ventas_product DESC;
```



	product_name	cant_ventas_product	price	colour
▶	riverlands north	60	\$169.96	#545454
	Winterfell	59	\$195.94	#1c1c1c
	Tarly Stark	56	\$9.24	#919191
	duel	54	\$59.80	#5b5b5b
	skywalker ewok sith	54	\$91.89	#7c7c7c
	jinn Winterfell	53	\$65.25	#bababa
	Disownelf riverlands the	52	\$122.86	#b7b7b7

Result 4 x

Output

Action Output

#	Time	Action	Message
✓ 1	21:35:28	SELECT product_name, COUNT(product_id) AS cant_ventas_product, price, colour FROM ...	26 row(s) returned

A continuació expongo la imagen del modelo de la base de datos final que he obtenido adicionando al modelo del nivel 1, los ejercicios del nivel 2 y nivel 3 (solución 1).

En el nivel 2, he creado la tabla estat\_targetes con una relación de muchos (estat\_targetes) a 1 con la tabla credit\_cards. La línea continua de la relación indica que es una tabla temporal.

En el nivel 3, he creado la tabla transactions\_products para poder relacionar la tabla products al modelo. Las relaciones creadas son las siguientes:

- de 1 a muchos, entre la tabla transactions y la tabla transactions\_products, es decir, 1 transacción puede contener varios productos.
- de 1 a muchos, entre la tabla transactions\_products y la tabla products, es decir, 1 producto puede ser vendido en varias transacciones.

