SC2207 Introduction to Databases

Lab 5:
Final Demonstration

Tutorial Group SCSZ Group 4

Group Members	Matriculation Number
ADRIAN CHUA PIN YU	U2222493K
BHUPATHIRAJU MIHIR VARMA	U2223870K
DARRELL MA WEI ZE	U2222176G
EUNICE LAI YING XUAN	U2240530D
TAN JING JIE	U2221344L

SQL DDL commands for Table Creation

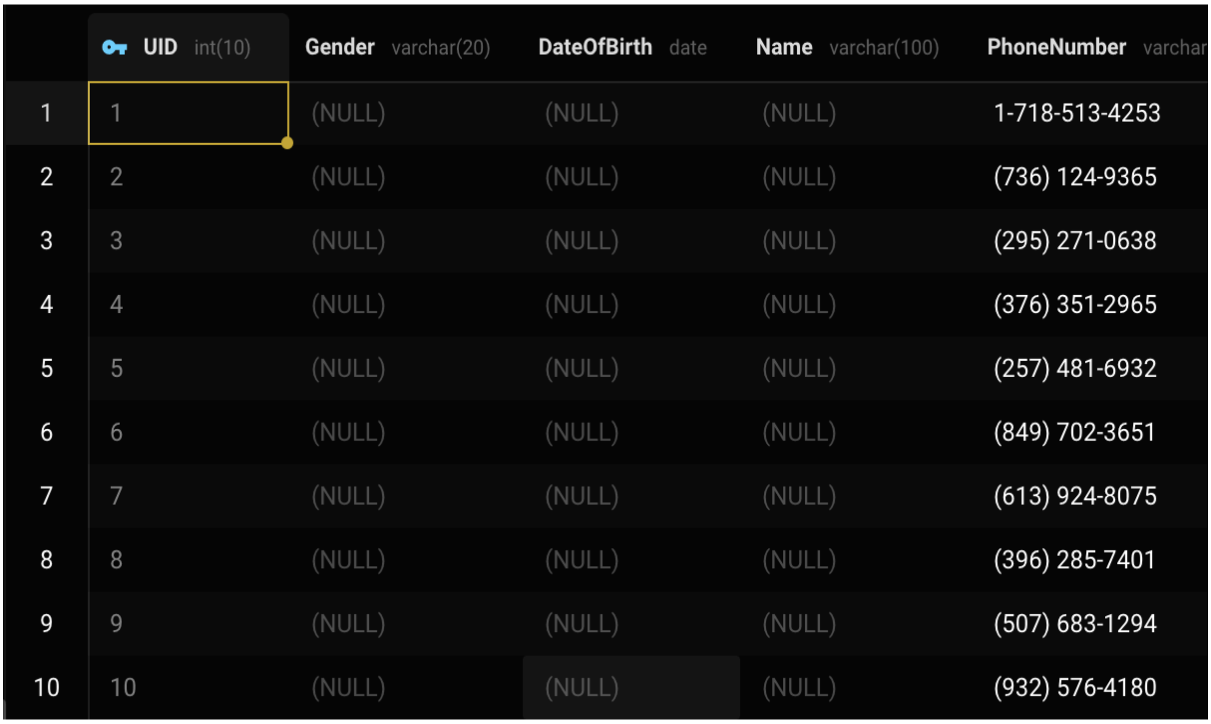
```
CREATE TABLE UserAccount (
   UID int NOT NULL IDENTITY(1, 1),
   Gender varchar(20),
   DateOfBirth date,
   Name varchar(100),
   PhoneNumber varchar(40) NOT NULL,
   PRIMARY KEY (UID)
);

CREATE TABLE RestaurantChain (
   RID int NOT NULL IDENTITY(1, 1),
   Address varchar(255) NOT NULL,
   PRIMARY KEY (RID)
);

CREATE TABLE MallManagementCompany (
   CID int NOT NULL IDENTITY(1, 1),
   Address varchar(255) NOT NULL,
   PRIMARY KEY (CID)
);

CREATE TABLE Mall (
   MID int NOT NULL IDENTITY(1, 1),
   CID int NOT NULL,
   Address varchar(255) NOT NULL,
   NumShops int NOT NULL,
   PRIMARY KEY (MID),
   FOREIGN KEY (CID) references MallManagementCompany(CID)
);

CREATE TABLE RestaurantOutlet (
   OID int NOT NULL IDENTITY(1, 1),
   RID int NOT NULL,
   MID int NOT NULL,
   PRIMARY KEY (OID),
   FOREIGN KEY (RID) references RestaurantChain(RID),
   FOREIGN KEY (MID) references Mall(MID)
);

CREATE TABLE UserDineRestaurant(
   UID int NOT NULL,
   OID int NOT NULL,
   AmountSpent money NOT NULL,
   DateTimeIn datetime NOT NULL,
   DateTimeOut datetime NOT NULL,
   PRIMARY KEY (UID, OID),
   FOREIGN KEY (UID) references UserAccount(UID),
   FOREIGN KEY (OID) references RestaurantOutlet(OID)
);

CREATE TABLE Shop (
   SID int NOT NULL IDENTITY(1, 1),
   Type varchar(40) NOT NULL,
   MID int NOT NULL,
   PRIMARY KEY (SID),
   FOREIGN KEY (MID) REFERENCES Mall(MID)
);

CREATE TABLE UserShopInMall (
   SID int NOT NULL,
   UID int NOT NULL,
   AmountSpent money NOT NULL,
   DateTimeIn datetime NOT NULL,
   DateTimeOut datetime NOT NULL,
   PRIMARY KEY (SID, UID, DateTimeIn, DateTimeOut),
   FOREIGN KEY (UID) references UserAccount(UID),
   FOREIGN KEY (SID) REFERENCES Shop(SID)
);

CREATE TABLE Recommendation (
   NID int NOT NULL IDENTITY(1, 1),
   DateIssued datetime NOT NULL DEFAULT (sysutcdatetime()),
   ExpiryDate datetime NOT NULL DEFAULT (sysutcdatetime()),
   MID int NOT NULL,
   PRIMARY KEY (NID),
   FOREIGN KEY (MID) references Mall(MID)
);

CREATE TABLE UserRelation (
   UID1 int NOT NULL,
   UID2 int NOT NULL,
   Type varchar(255) NOT NULL,
   PRIMARY KEY (UID1, UID2),
   FOREIGN KEY (UID1) references UserAccount(UID),
   FOREIGN KEY (UID2) references UserAccount(UID)
);

CREATE TABLE UserAccountUseRecommendation (
   UID int NOT NULL,
   NID int NOT NULL,
   PRIMARY KEY (UID, NID),
   FOREIGN KEY (UID) references UserAccount(UID),
   FOREIGN KEY (NID) references Recommendation(NID)
);

-- Voucher
CREATE TABLE Voucher (
   VID INT NOT NULL IDENTITY(1, 1),
   DateIssued DATE,
   VDescription VARCHAR(100),
   VStatus VARCHAR(100),
   ExpiryDate DATE,
   PRIMARY KEY (VID)
);

CREATE TABLE PurchaseVoucher (
```

```sql
   VID INT NOT NULL,
   PurchaseDiscount MONEY,
   Date_Time DATETIME,
   UID INT,
   PRIMARY KEY(VID),
   FOREIGN KEY(VID) REFERENCES Voucher(VID),
   FOREIGN KEY(UID) REFERENCES UserAccount(UID)
);

CREATE TABLE DineVoucher (
   VID INT NOT NULL,
   CashDiscount MONEY,
   Date_Time DATETIME,
   UID INT,
   PRIMARY KEY (VID),
   FOREIGN KEY(VID) REFERENCES Voucher(VID),
   FOREIGN KEY(UID) REFERENCES UserAccount(UID)
);

CREATE TABLE GroupVoucher (
   VID INT NOT NULL,
   GroupSize INT,
   GroupDiscount MONEY,
   Date_Time DATETIME,
   UID INT,
   PRIMARY KEY (VID),
   FOREIGN KEY(VID) REFERENCES Voucher(VID),
   FOREIGN KEY(UID) REFERENCES UserAccount(UID)
);

CREATE TABLE PackageVoucher (
   VID INT NOT NULL,
   PackageDiscount MONEY,
   PRIMARY KEY (VID),
   FOREIGN KEY(VID) REFERENCES Voucher(VID),
);

-- Day Package
CREATE TABLE DayPackage (
   DID int NOT NULL IDENTITY(1, 1),
   Description varchar(255) NOT NULL,
   VID int NOT NULL,
   PRIMARY KEY (DID),
   FOREIGN KEY (VID) references PackageVoucher(VID)
);

CREATE TABLE UserUsesDayPackage (
   UID int,
   DID int,
   PRIMARY KEY (UID, DID),
   FOREIGN KEY (DID) REFERENCES DayPackage(DID),
   FOREIGN KEY (UID) REFERENCES UserAccount(UID)
);
```

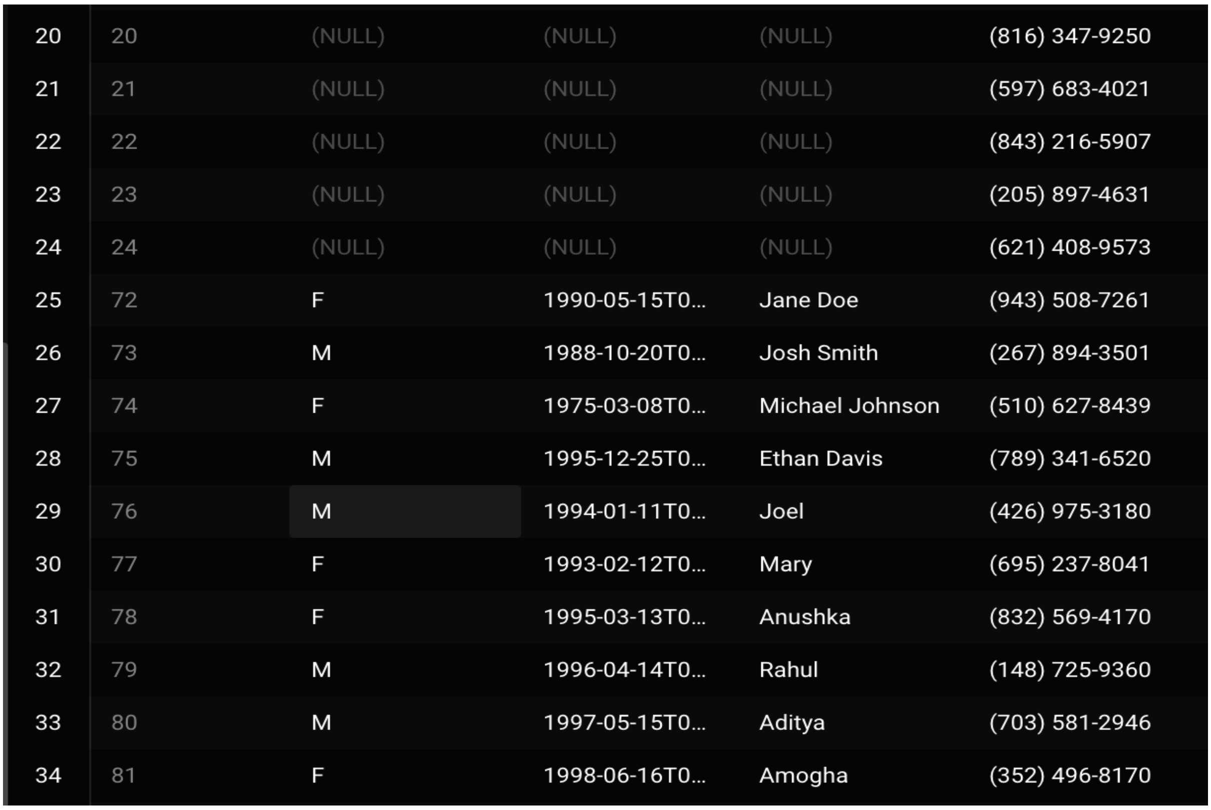
```
CREATE TABLE RestaurantOutletHasDayPackage (
   OID int NOT NULL,
   DID int NOT NULL,
   PRIMARY KEY (OID, DID),
   FOREIGN KEY (OID) references RestaurantOutlet(OID),
   FOREIGN KEY (DID) references DayPackage(DID)
);

CREATE TABLE MallHasDayPackage (
   MID int NOT NULL,
   DID int NOT NULL,
   PRIMARY KEY (MID, DID),
   FOREIGN KEY (MID) references Mall(MID),
   FOREIGN KEY (DID) references DayPackage(DID)
);

-- Complaints
CREATE TABLE Complaint (
   CID int NOT NULL IDENTITY(1, 1),
   UID int NOT NULL,
   Text VARCHAR(255) NOT NULL,
   Status VARCHAR(100) NOT NULL,
   FilledDateTime DATETIME NOT NULL,
   PRIMARY KEY (CID),
   FOREIGN KEY (UID) references UserAccount(UID)
);

CREATE TABLE ComplaintsOnRestaurant (
   CID int NOT NULL,
   OID int NOT NULL,
   PRIMARY KEY (CID),
   FOREIGN KEY (CID) references Complaint(CID),
   FOREIGN KEY (OID) references RestaurantOutlet(OID)
);

CREATE TABLE ComplaintsOnShop (
   CID int NOT NULL,
   SID int NOT NULL,
   PRIMARY KEY (CID),
   FOREIGN KEY (CID) references Complaint(CID),
   FOREIGN KEY (SID) references Shop(SID)
);
```

Table Records (Printout of Table)

UserAccount:

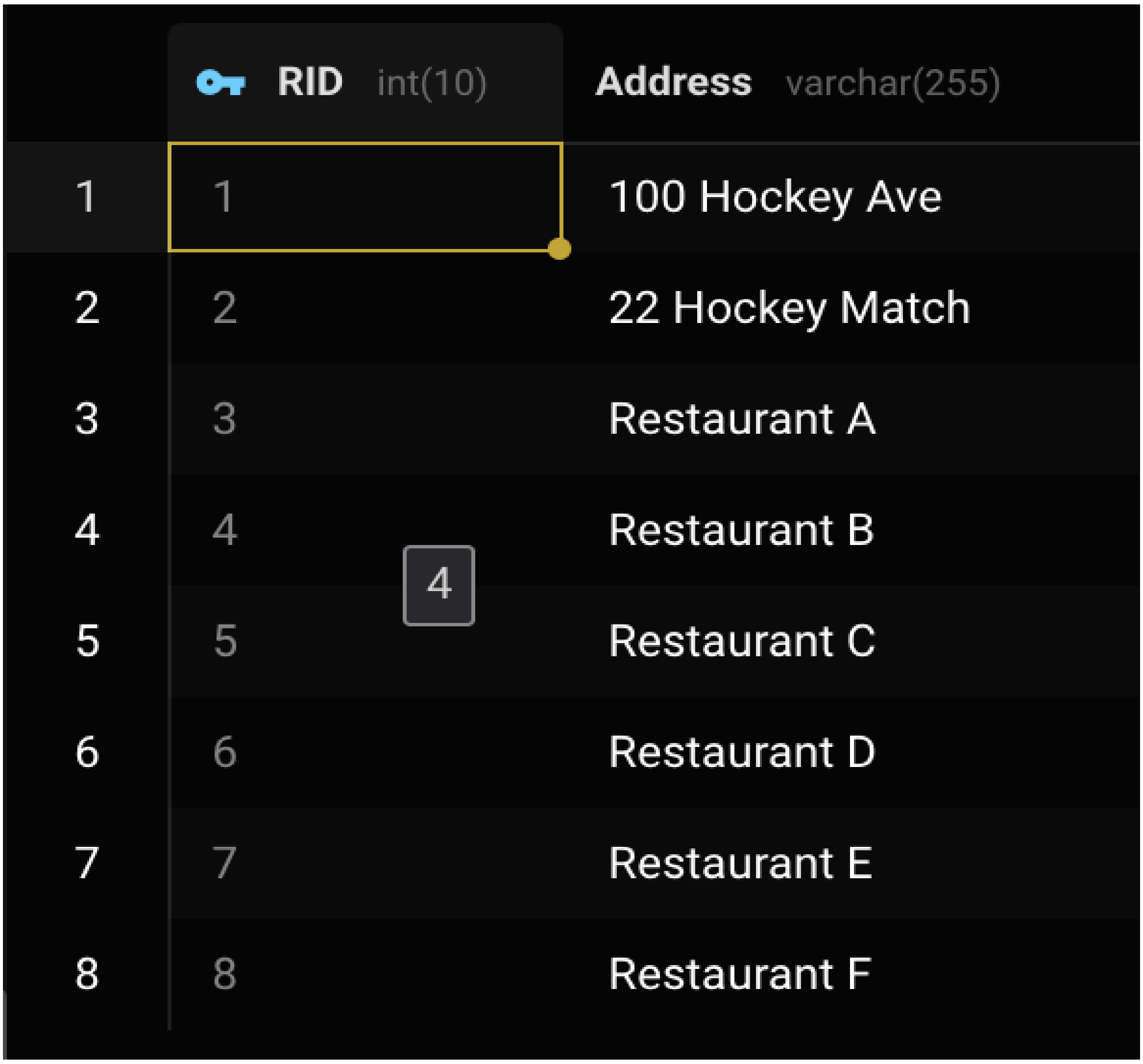


	UID int(10)	Gender varchar(20)	DateOfBirth date	Name varchar(100)	PhoneNumber varchar
1	1	(NULL)	(NULL)	(NULL)	1-718-513-4253
2	2	(NULL)	(NULL)	(NULL)	(736) 124-9365
3	3	(NULL)	(NULL)	(NULL)	(295) 271-0638
4	4	(NULL)	(NULL)	(NULL)	(376) 351-2965
5	5	(NULL)	(NULL)	(NULL)	(257) 481-6932
6	6	(NULL)	(NULL)	(NULL)	(849) 702-3651
7	7	(NULL)	(NULL)	(NULL)	(613) 924-8075
8	8	(NULL)	(NULL)	(NULL)	(396) 285-7401
9	9	(NULL)	(NULL)	(NULL)	(507) 683-1294
10	10	(NULL)	(NULL)	(NULL)	(932) 576-4180



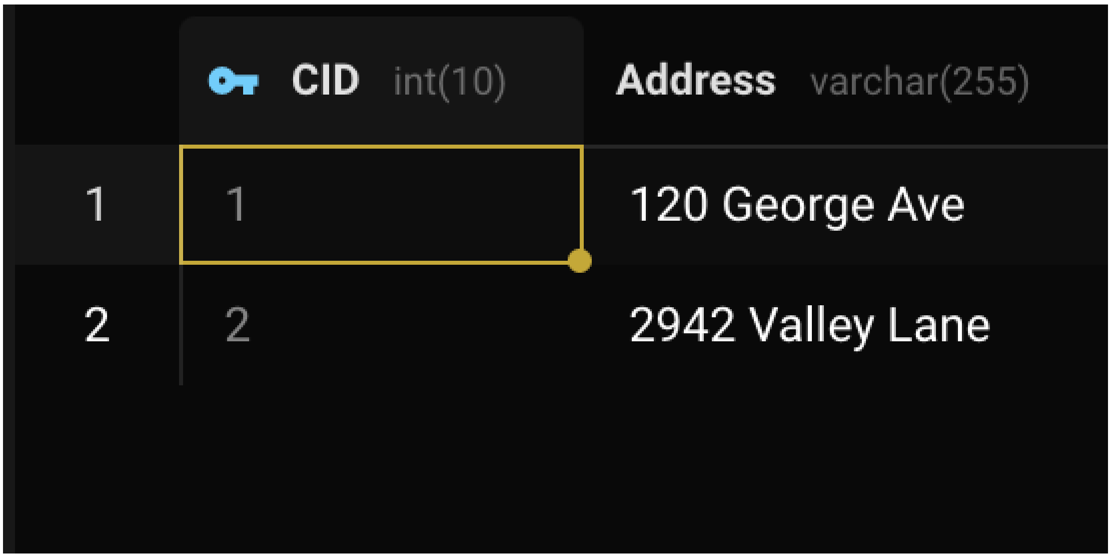
20	20	(NULL)	(NULL)	(NULL)	(816) 347-9250
21	21	(NULL)	(NULL)	(NULL)	(597) 683-4021
22	22	(NULL)	(NULL)	(NULL)	(843) 216-5907
23	23	(NULL)	(NULL)	(NULL)	(205) 897-4631
24	24	(NULL)	(NULL)	(NULL)	(621) 408-9573
25	72	F	1990-05-15T0...	Jane Doe	(943) 508-7261
26	73	M	1988-10-20T0...	Josh Smith	(267) 894-3501
27	74	F	1975-03-08T0...	Michael Johnson	(510) 627-8439
28	75	M	1995-12-25T0...	Ethan Davis	(789) 341-6520
29	76	M	1994-01-11T0...	Joel	(426) 975-3180
30	77	F	1993-02-12T0...	Mary	(695) 237-8041
31	78	F	1995-03-13T0...	Anushka	(832) 569-4170
32	79	M	1996-04-14T0...	Rahul	(148) 725-9360
33	80	M	1997-05-15T0...	Aditya	(703) 581-2946
34	81	F	1998-06-16T0...	Amogha	(352) 496-8170

RestaurantChain:



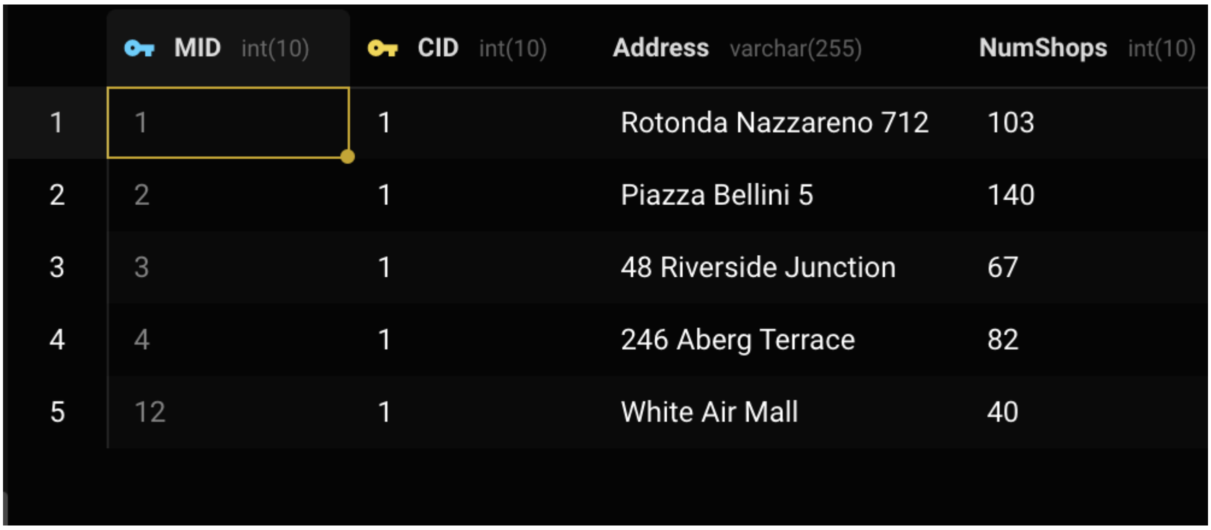
	🔑 RID int(10)	Address varchar(255)
1	1	100 Hockey Ave
2	2	22 Hockey Match
3	3	Restaurant A
4	4	Restaurant B
5	5	Restaurant C
6	6	Restaurant D
7	7	Restaurant E
8	8	Restaurant F

MallManagementCompany:



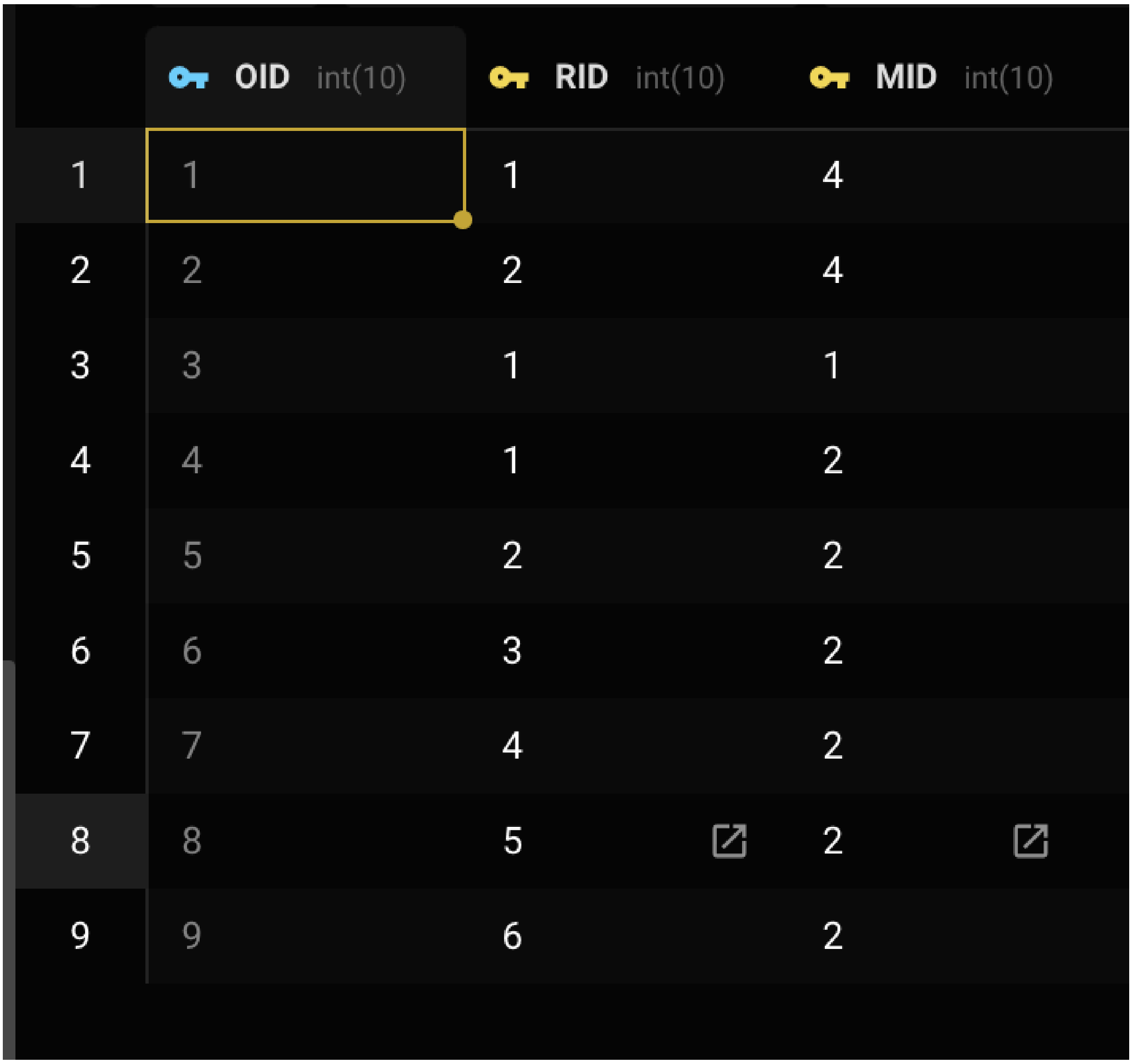
	🔑 CID int(10)	Address varchar(255)
1	1	120 George Ave
2	2	2942 Valley Lane

Mall:



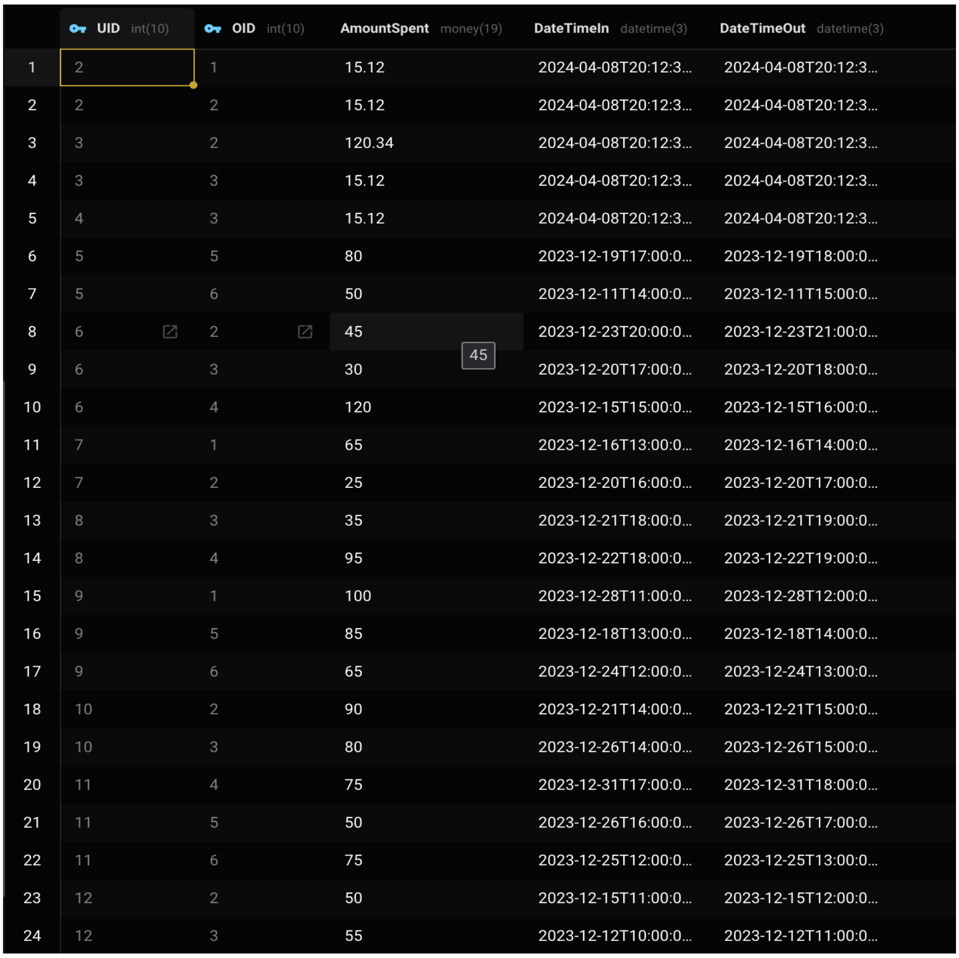
	MID int(10)	CID int(10)	Address varchar(255)	NumShops int(10)
1	1	1	Rotonda Nazzareno 712	103
2	2	1	Piazza Bellini 5	140
3	3	1	48 Riverside Junction	67
4	4	1	246 Aberg Terrace	82
5	12	1	White Air Mall	40

RestaurantOutlet:



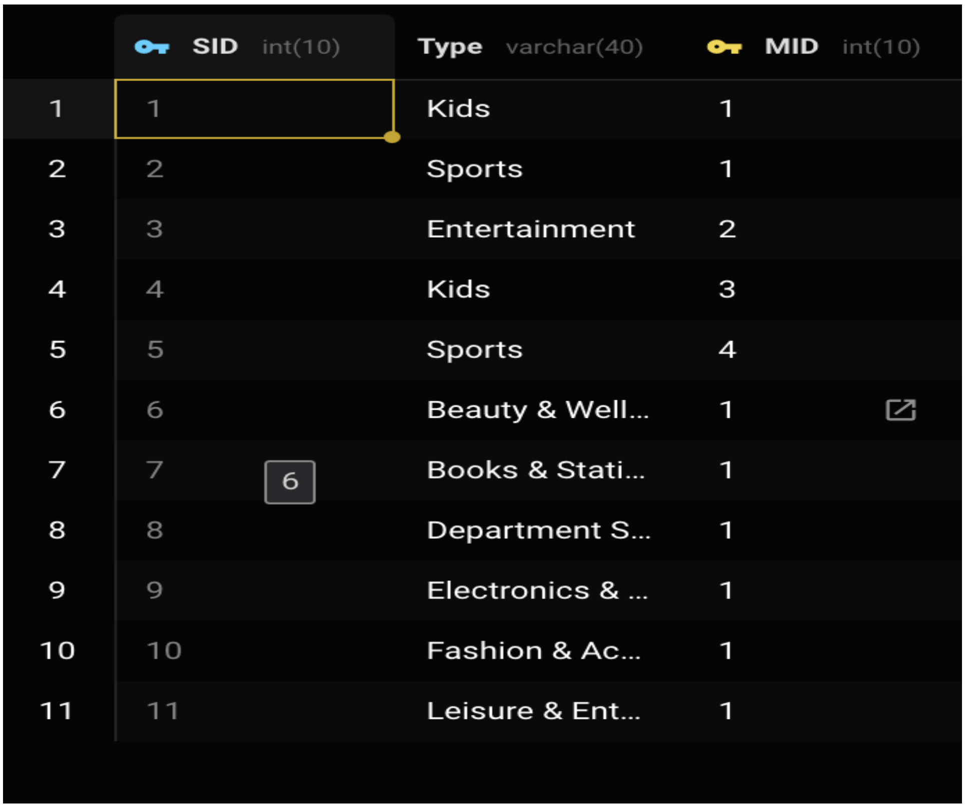
	OID int(10)	RID int(10)	MID int(10)
1	1	1	4
2	2	2	4
3	3	1	1
4	4	1	2
5	5	2	2
6	6	3	2
7	7	4	2
8	8	5	2
9	9	6	2

UserDineRestaurant:



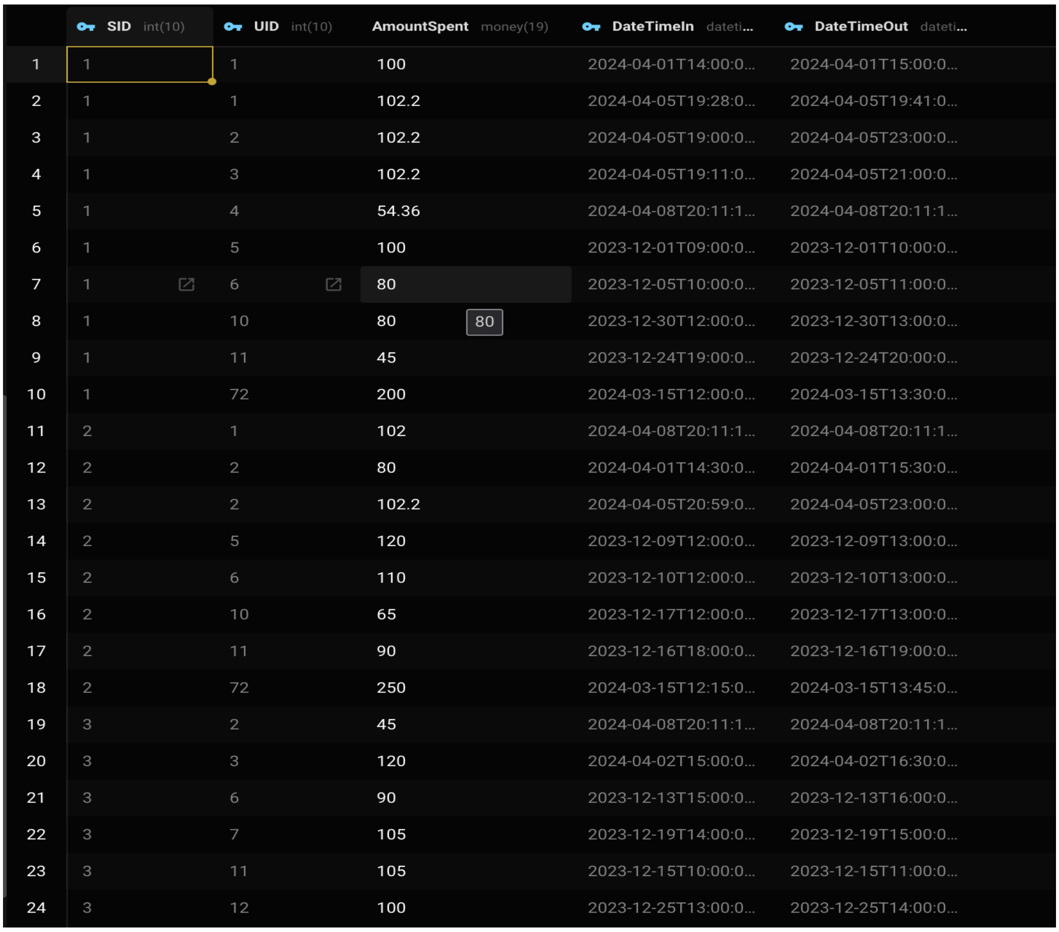
	UID int(10)	OID int(10)	AmountSpent money(19)	DateTimeIn datetime(3)	DateTimeOut datetime(3)
1	2	1	15.12	2024-04-08T20:12:3...	2024-04-08T20:12:3...
2	2	2	15.12	2024-04-08T20:12:3...	2024-04-08T20:12:3...
3	3	2	120.34	2024-04-08T20:12:3...	2024-04-08T20:12:3...
4	3	3	15.12	2024-04-08T20:12:3...	2024-04-08T20:12:3...
5	4	3	15.12	2024-04-08T20:12:3...	2024-04-08T20:12:3...
6	5	5	80	2023-12-19T17:00:0...	2023-12-19T18:00:0...
7	5	6	50	2023-12-11T14:00:0...	2023-12-11T15:00:0...
8	6	2	45	2023-12-23T20:00:0...	2023-12-23T21:00:0...
9	6	3	30	2023-12-20T17:00:0...	2023-12-20T18:00:0...
10	6	4	120	2023-12-15T15:00:0...	2023-12-15T16:00:0...
11	7	1	65	2023-12-16T13:00:0...	2023-12-16T14:00:0...
12	7	2	25	2023-12-20T16:00:0...	2023-12-20T17:00:0...
13	8	3	35	2023-12-21T18:00:0...	2023-12-21T19:00:0...
14	8	4	95	2023-12-22T18:00:0...	2023-12-22T19:00:0...
15	9	1	100	2023-12-28T11:00:0...	2023-12-28T12:00:0...
16	9	5	85	2023-12-18T13:00:0...	2023-12-18T14:00:0...
17	9	6	65	2023-12-24T12:00:0...	2023-12-24T13:00:0...
18	10	2	90	2023-12-21T14:00:0...	2023-12-21T15:00:0...
19	10	3	80	2023-12-26T14:00:0...	2023-12-26T15:00:0...
20	11	4	75	2023-12-31T17:00:0...	2023-12-31T18:00:0...
21	11	5	50	2023-12-26T16:00:0...	2023-12-26T17:00:0...
22	11	6	75	2023-12-25T12:00:0...	2023-12-25T13:00:0...
23	12	2	50	2023-12-15T11:00:0...	2023-12-15T12:00:0...
24	12	3	55	2023-12-12T10:00:0...	2023-12-12T11:00:0...

Shop:



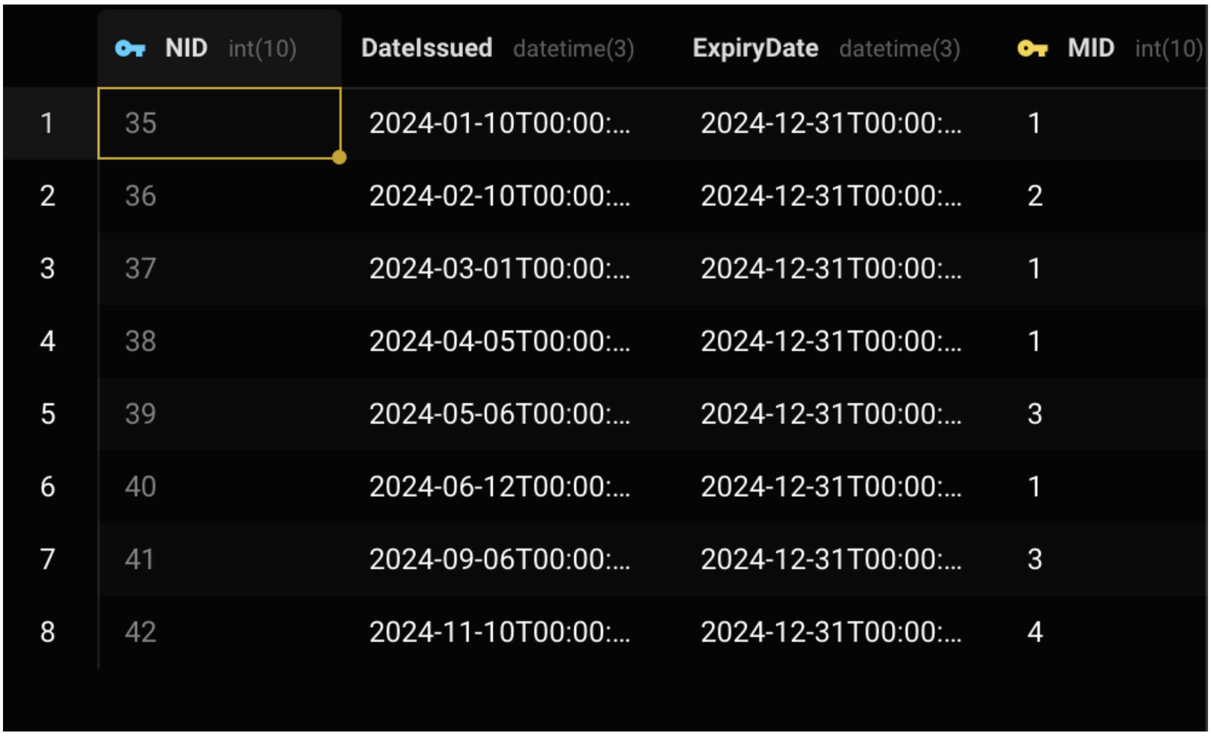
	SID int(10)	Type varchar(40)	MID int(10)
1	1	Kids	1
2	2	Sports	1
3	3	Entertainment	2
4	4	Kids	3
5	5	Sports	4
6	6	Beauty & Well...	1
7	7	Books & Stati...	1
8	8	Department S...	1
9	9	Electronics & ...	1
10	10	Fashion & Ac...	1
11	11	Leisure & Ent...	1

UserShopInMall:



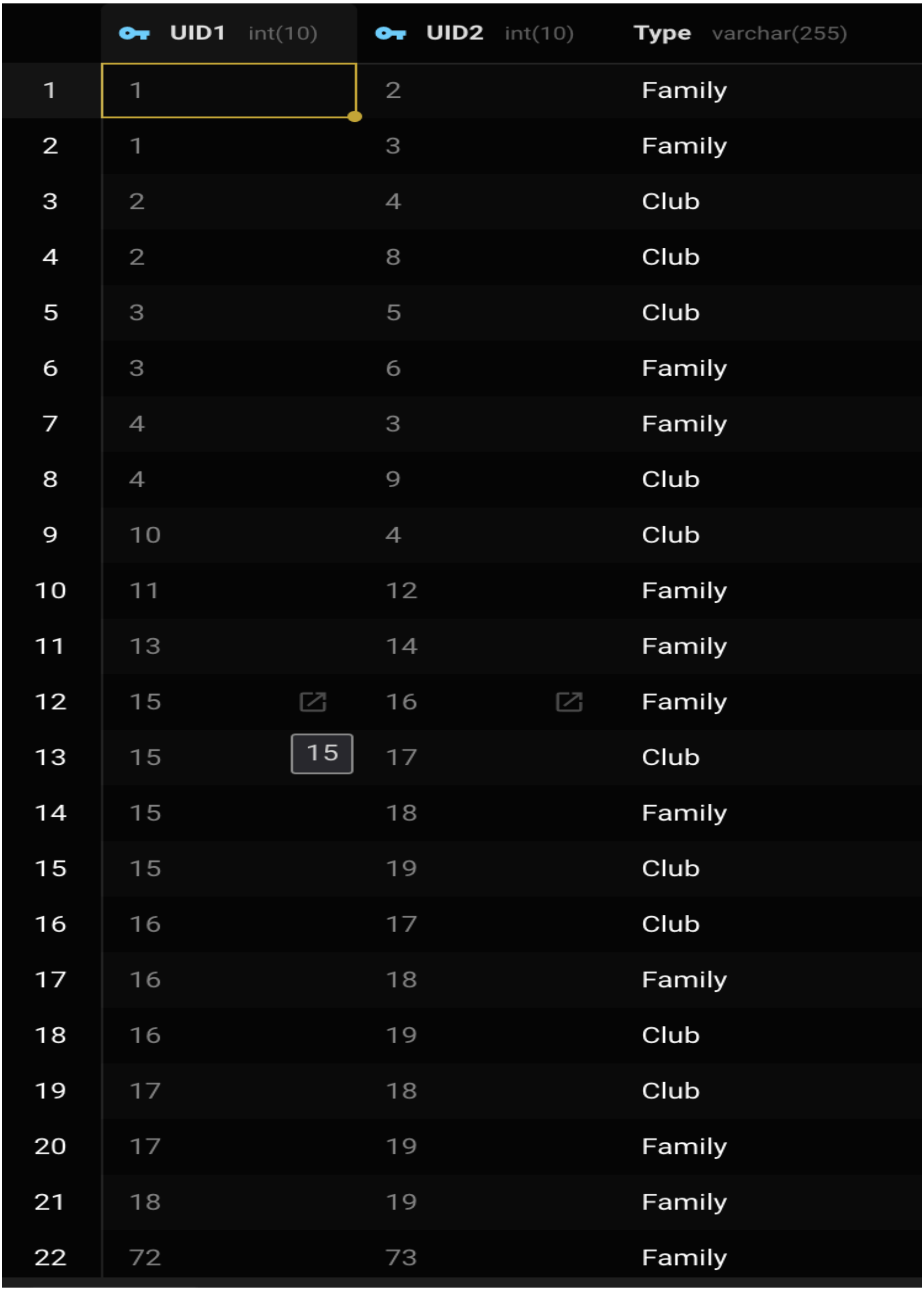
	SID int(10)	UID int(10)	AmountSpent money(19)	DateTimeIn dateti...	DateTimeOut dateti...
1	1	1	100	2024-04-01T14:00:0...	2024-04-01T15:00:0...
2	1	1	102.2	2024-04-05T19:28:0...	2024-04-05T19:41:0...
3	1	2	102.2	2024-04-05T19:00:0...	2024-04-05T23:00:0...
4	1	3	102.2	2024-04-05T19:11:0...	2024-04-05T21:00:0...
5	1	4	54.36	2024-04-08T20:11:1...	2024-04-08T20:11:1...
6	1	5	100	2023-12-01T09:00:0...	2023-12-01T10:00:0...
7	1	6	80	2023-12-05T10:00:0...	2023-12-05T11:00:0...
8	1	10	80	2023-12-30T12:00:0...	2023-12-30T13:00:0...
9	1	11	45	2023-12-24T19:00:0...	2023-12-24T20:00:0...
10	1	72	200	2024-03-15T12:00:0...	2024-03-15T13:30:0...
11	2	1	102	2024-04-08T20:11:1...	2024-04-08T20:11:1...
12	2	2	80	2024-04-01T14:30:0...	2024-04-01T15:30:0...
13	2	2	102.2	2024-04-05T20:59:0...	2024-04-05T23:00:0...
14	2	5	120	2023-12-09T12:00:0...	2023-12-09T13:00:0...
15	2	6	110	2023-12-10T12:00:0...	2023-12-10T13:00:0...
16	2	10	65	2023-12-17T12:00:0...	2023-12-17T13:00:0...
17	2	11	90	2023-12-16T18:00:0...	2023-12-16T19:00:0...
18	2	72	250	2024-03-15T12:15:0...	2024-03-15T13:45:0...
19	3	2	45	2024-04-08T20:11:1...	2024-04-08T20:11:1...
20	3	3	120	2024-04-02T15:00:0...	2024-04-02T16:30:0...
21	3	6	90	2023-12-13T15:00:0...	2023-12-13T16:00:0...
22	3	7	105	2023-12-19T14:00:0...	2023-12-19T15:00:0...
23	3	11	105	2023-12-15T10:00:0...	2023-12-15T11:00:0...
24	3	12	100	2023-12-25T13:00:0...	2023-12-25T14:00:0...

Recommendation:



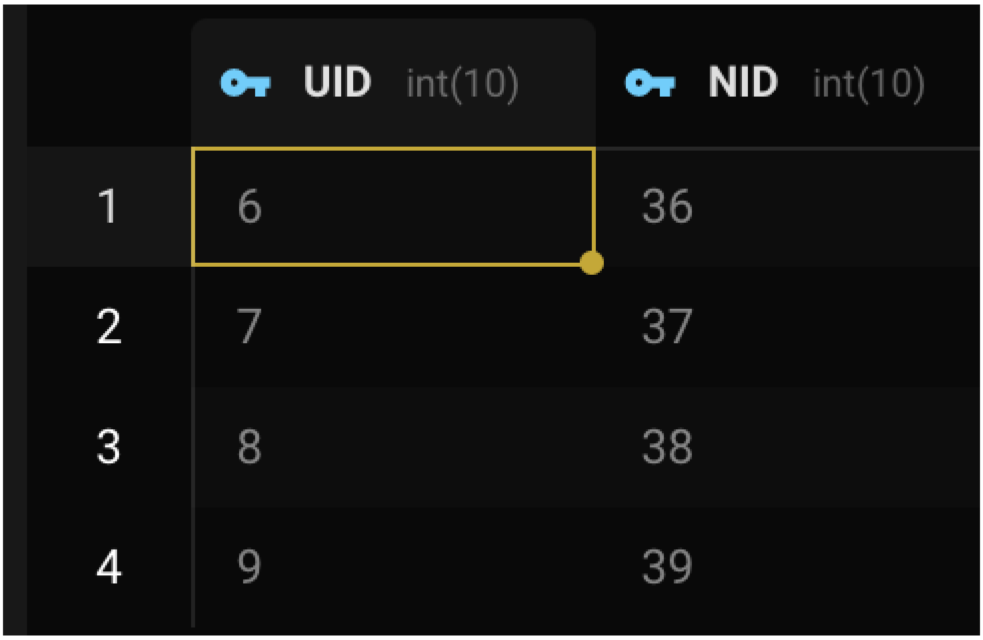
	NID int(10)	DateIssued datetime(3)	ExpiryDate datetime(3)	MID int(10)
1	35	2024-01-10T00:00:...	2024-12-31T00:00:...	1
2	36	2024-02-10T00:00:...	2024-12-31T00:00:...	2
3	37	2024-03-01T00:00:...	2024-12-31T00:00:...	1
4	38	2024-04-05T00:00:...	2024-12-31T00:00:...	1
5	39	2024-05-06T00:00:...	2024-12-31T00:00:...	3
6	40	2024-06-12T00:00:...	2024-12-31T00:00:...	1
7	41	2024-09-06T00:00:...	2024-12-31T00:00:...	3
8	42	2024-11-10T00:00:...	2024-12-31T00:00:...	4

UserRelation:



	UID1 int(10)	UID2 int(10)	Type varchar(255)
1	1	2	Family
2	1	3	Family
3	2	4	Club
4	2	8	Club
5	3	5	Club
6	3	6	Family
7	4	3	Family
8	4	9	Club
9	10	4	Club
10	11	12	Family
11	13	14	Family
12	15	16	Family
13	15	17	Club
14	15	18	Family
15	15	19	Club
16	16	17	Club
17	16	18	Family
18	16	19	Club
19	17	18	Club
20	17	19	Family
21	18	19	Family
22	72	73	Family

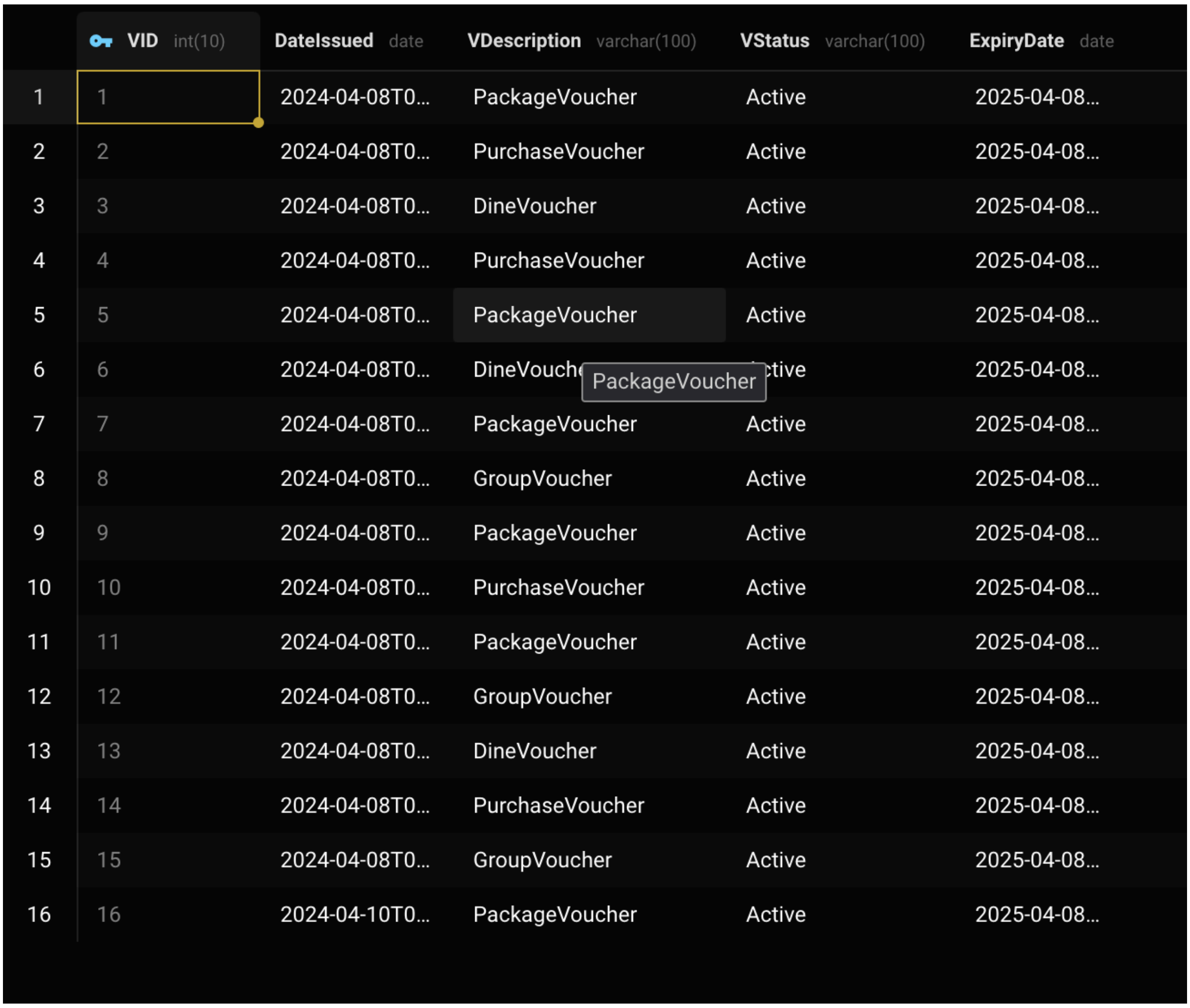
UserAccountUseRecommendation:



	UID int(10)	NID int(10)
1	6	36
2	7	37
3	8	38
4	9	39

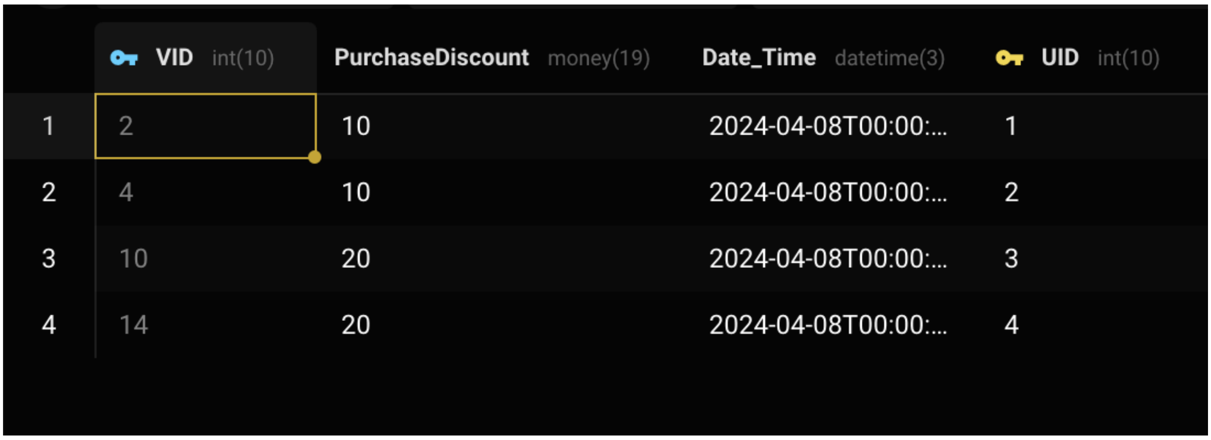
-- Voucher

Voucher:



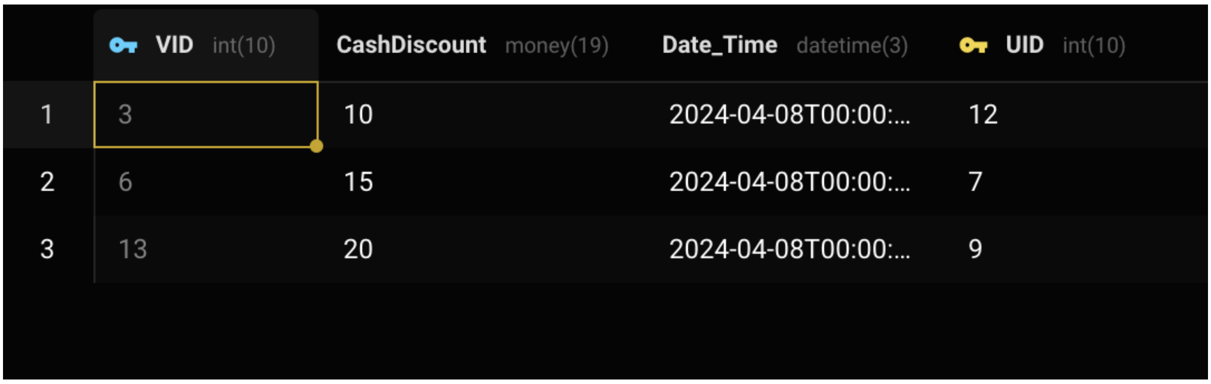
	VID int(10)	DateIssued date	VDescription varchar(100)	VStatus varchar(100)	ExpiryDate date
1	1	2024-04-08T0...	PackageVoucher	Active	2025-04-08...
2	2	2024-04-08T0...	PurchaseVoucher	Active	2025-04-08...
3	3	2024-04-08T0...	DineVoucher	Active	2025-04-08...
4	4	2024-04-08T0...	PurchaseVoucher	Active	2025-04-08...
5	5	2024-04-08T0...	PackageVoucher	Active	2025-04-08...
6	6	2024-04-08T0...	DineVouche	ctive	2025-04-08...
7	7	2024-04-08T0...	PackageVoucher	Active	2025-04-08...
8	8	2024-04-08T0...	GroupVoucher	Active	2025-04-08...
9	9	2024-04-08T0...	PackageVoucher	Active	2025-04-08...
10	10	2024-04-08T0...	PurchaseVoucher	Active	2025-04-08...
11	11	2024-04-08T0...	PackageVoucher	Active	2025-04-08...
12	12	2024-04-08T0...	GroupVoucher	Active	2025-04-08...
13	13	2024-04-08T0...	DineVoucher	Active	2025-04-08...
14	14	2024-04-08T0...	PurchaseVoucher	Active	2025-04-08...
15	15	2024-04-08T0...	GroupVoucher	Active	2025-04-08...
16	16	2024-04-10T0...	PackageVoucher	Active	2025-04-08...

PurchaseVoucher:



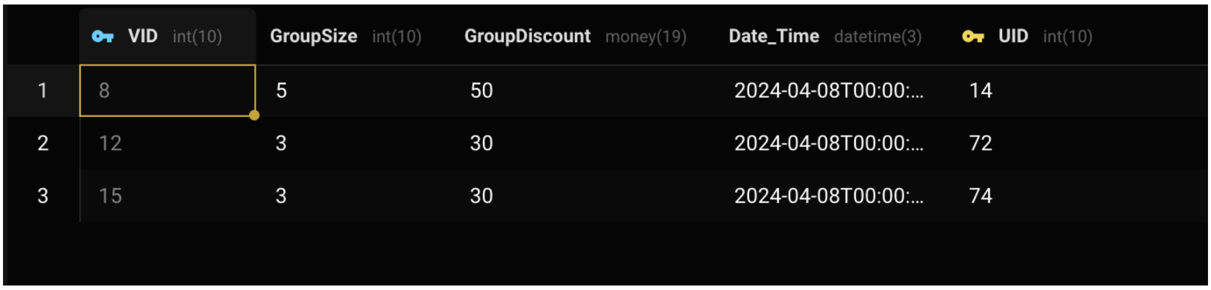
	VID int(10)	PurchaseDiscount money(19)	Date_Time datetime(3)	UID int(10)
1	2	10	2024-04-08T00:00:...	1
2	4	10	2024-04-08T00:00:...	2
3	10	20	2024-04-08T00:00:...	3
4	14	20	2024-04-08T00:00:...	4

DineVoucher:



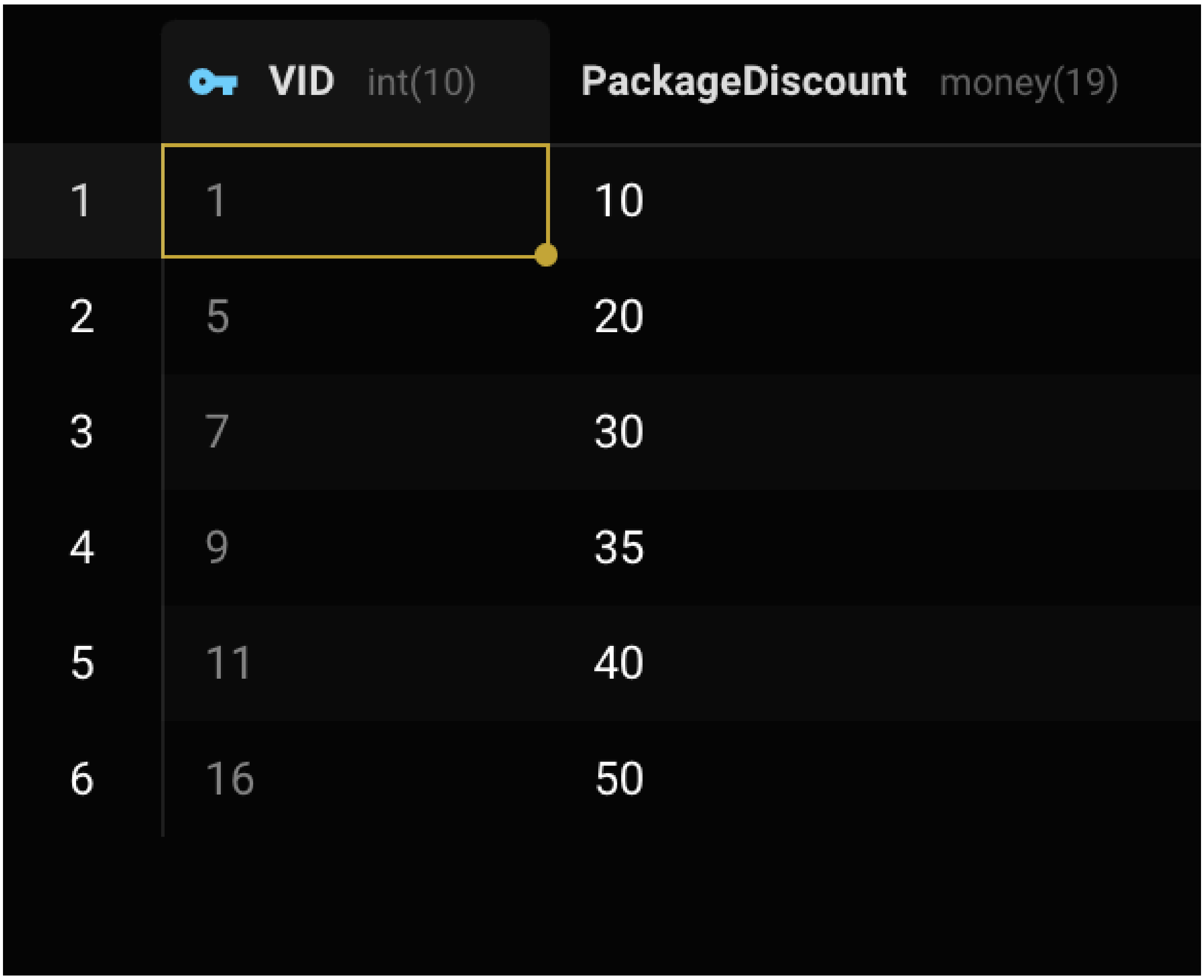
	VID int(10)	CashDiscount money(19)	Date_Time datetime(3)	UID int(10)
1	3	10	2024-04-08T00:00:...	12
2	6	15	2024-04-08T00:00:...	7
3	13	20	2024-04-08T00:00:...	9

GroupVoucher:



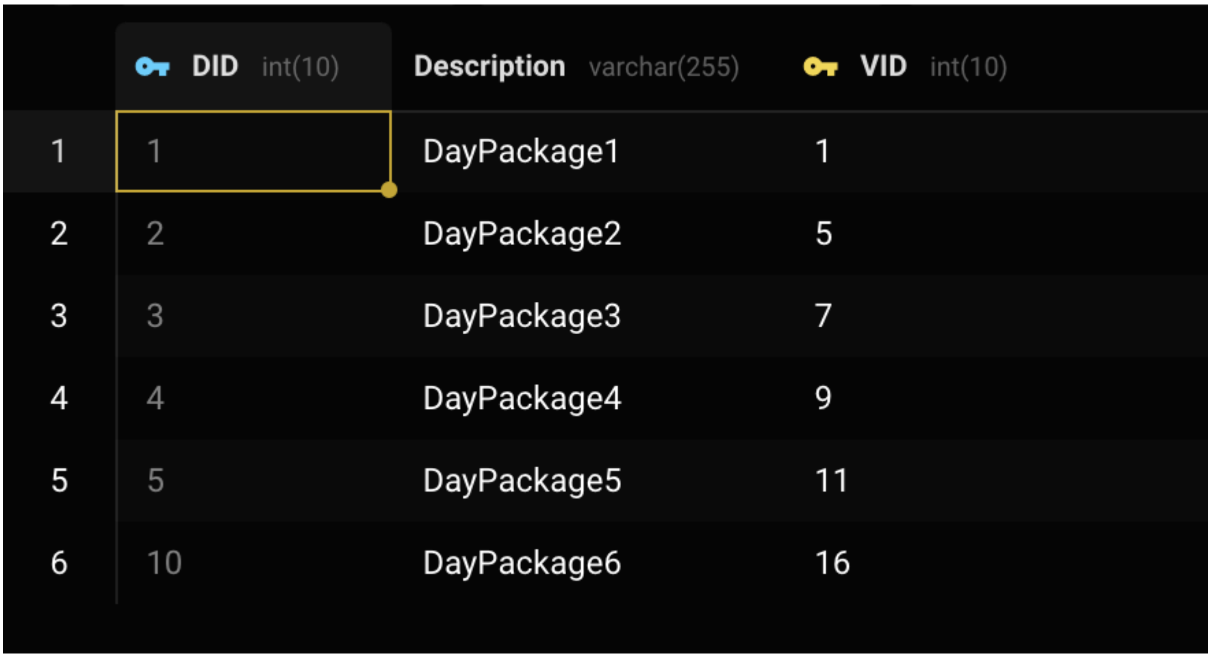
	VID int(10)	GroupSize int(10)	GroupDiscount money(19)	Date_Time datetime(3)	UID int(10)
1	8	5	50	2024-04-08T00:00:...	14
2	12	3	30	2024-04-08T00:00:...	72
3	15	3	30	2024-04-08T00:00:...	74

```
PackageVoucher:
```



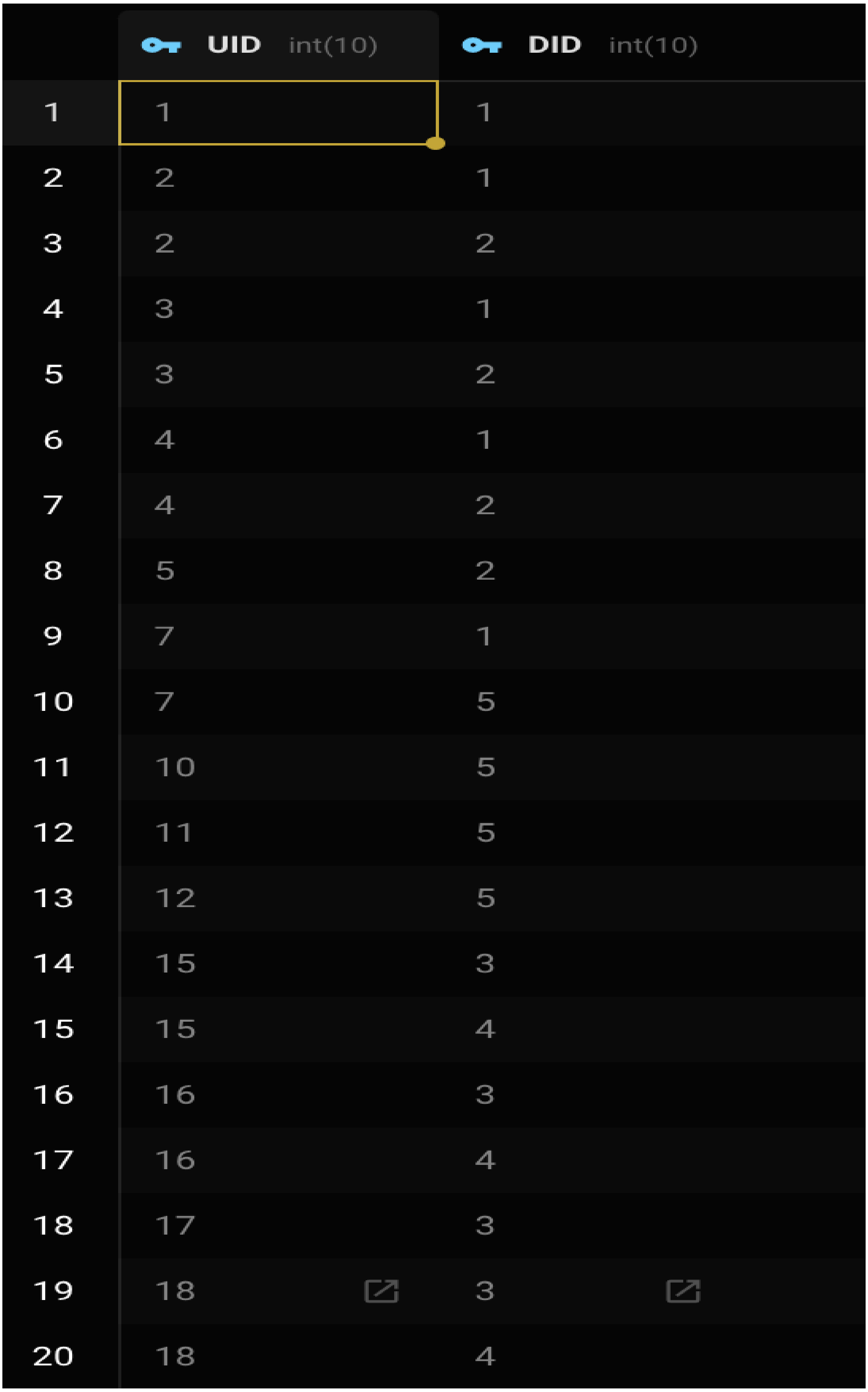
	🔑 VID int(10)	PackageDiscount money(19)
1	1	10
2	5	20
3	7	30
4	9	35
5	11	40
6	16	50

```
-- Day Package
DayPackage:
```



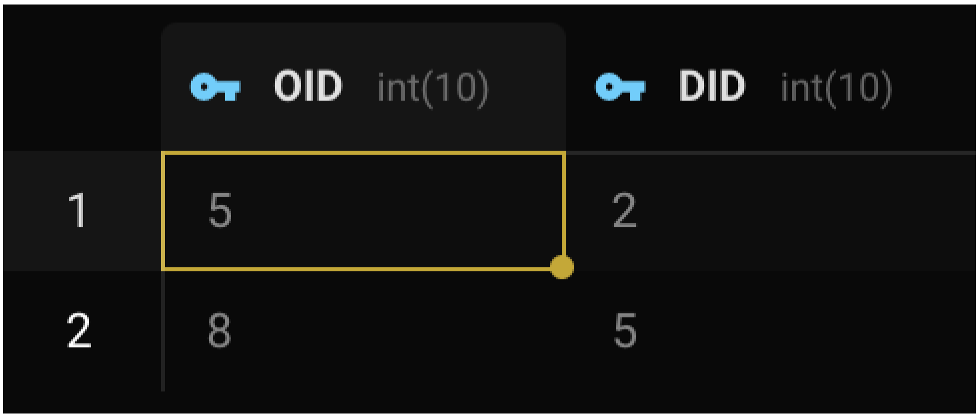
	🔑 DID int(10)	Description varchar(255)	🔑 VID int(10)
1	1	DayPackage1	1
2	2	DayPackage2	5
3	3	DayPackage3	7
4	4	DayPackage4	9
5	5	DayPackage5	11
6	10	DayPackage6	16

UserUsesDayPackage (First 20 records):



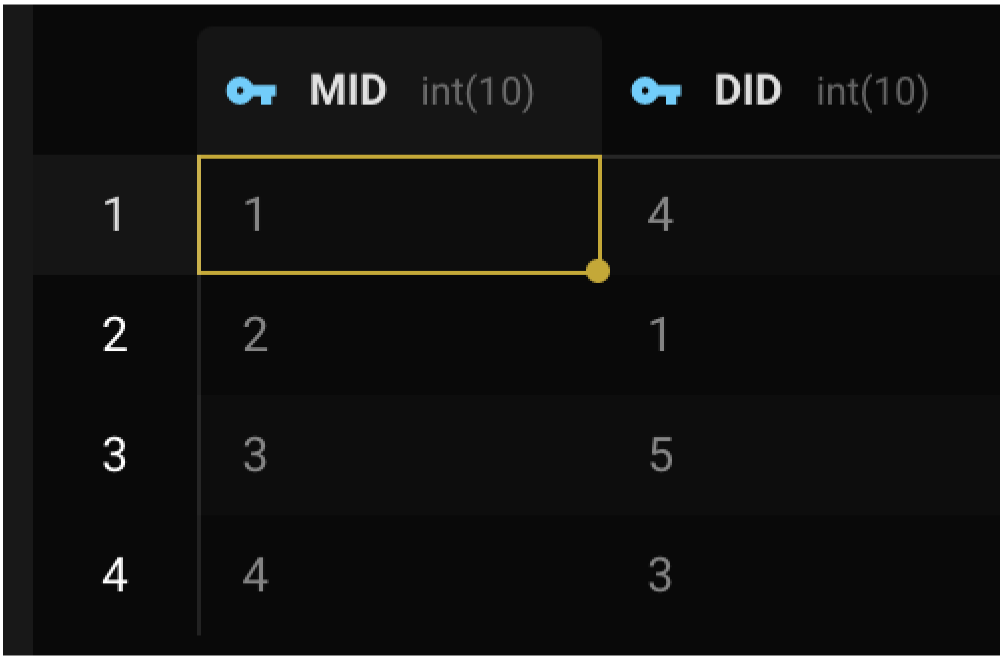
	UID int(10)	DID int(10)
1	1	1
2	2	1
3	2	2
4	3	1
5	3	2
6	4	1
7	4	2
8	5	2
9	7	1
10	7	5
11	10	5
12	11	5
13	12	5
14	15	3
15	15	4
16	16	3
17	16	4
18	17	3
19	18	3
20	18	4

RestaurantOutletHasDayPackage:



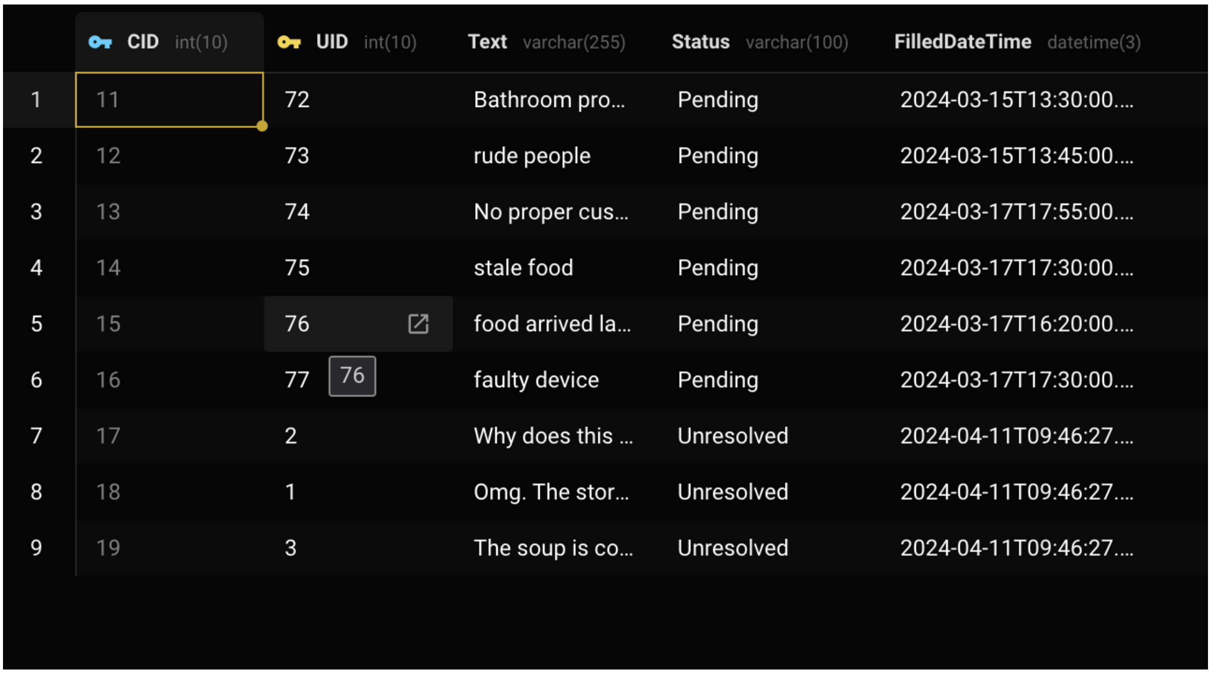
	OID int(10)	DID int(10)
1	5	2
2	8	5

MallHasDayPackage:



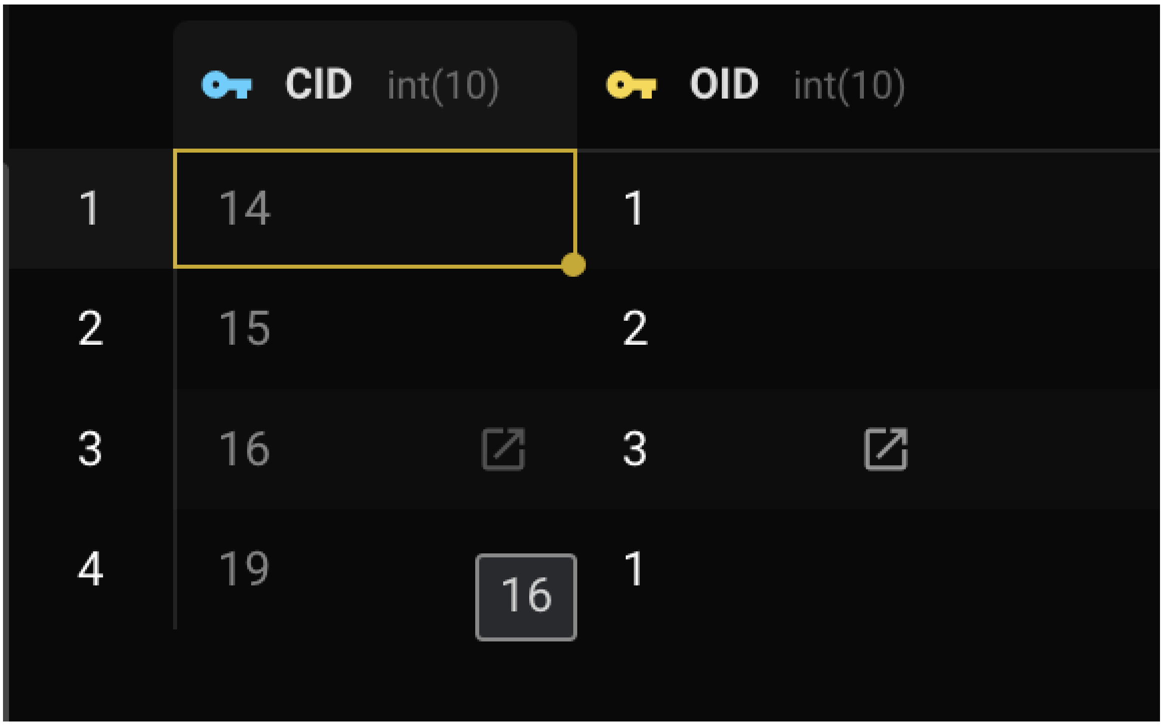
	MID int(10)	DID int(10)
1	1	4
2	2	1
3	3	5
4	4	3

```
-- Complaints
Complaint:
```



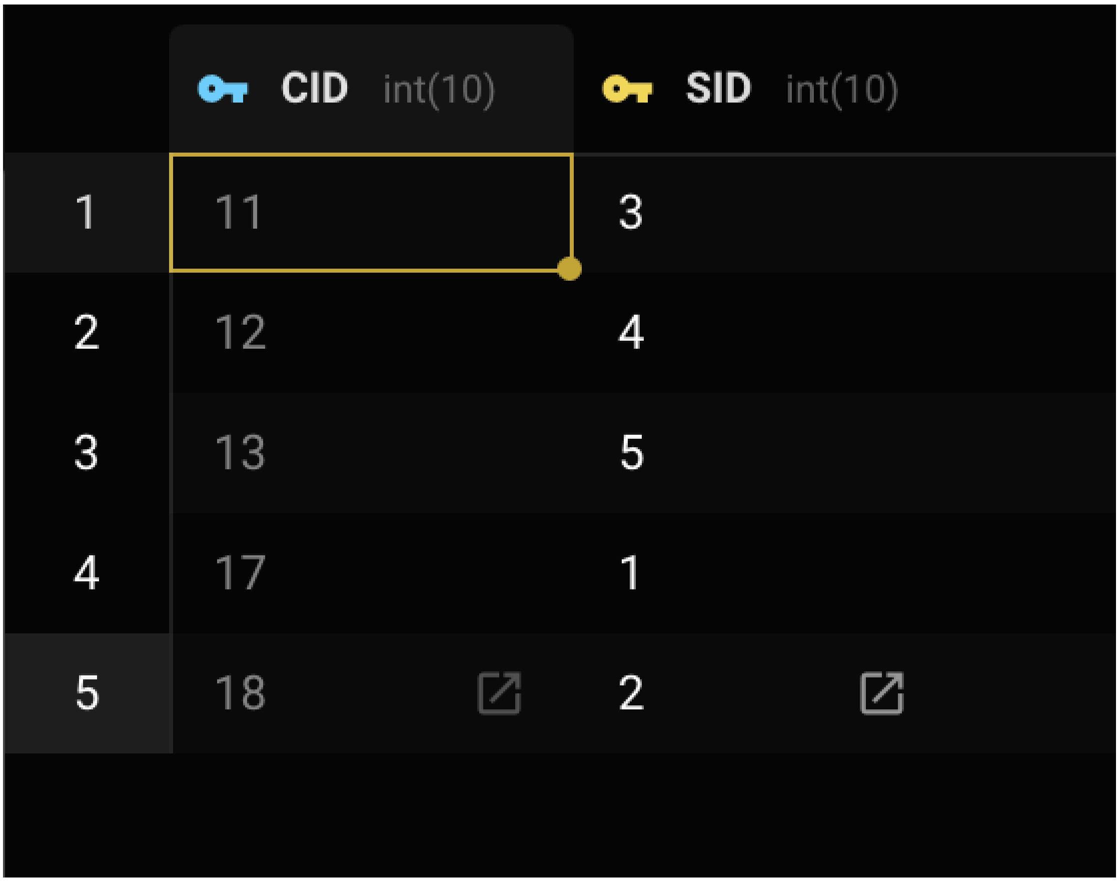
	CID int(10)	UID int(10)	Text varchar(255)	Status varchar(100)	FilledDateTime datetime(3)
1	11	72	Bathroom pro...	Pending	2024-03-15T13:30:00....
2	12	73	rude people	Pending	2024-03-15T13:45:00....
3	13	74	No proper cus...	Pending	2024-03-17T17:55:00....
4	14	75	stale food	Pending	2024-03-17T17:30:00....
5	15	76	food arrived la...	Pending	2024-03-17T16:20:00....
6	16	77	faulty device	Pending	2024-03-17T17:30:00....
7	17	2	Why does this ...	Unresolved	2024-04-11T09:46:27....
8	18	1	Omg. The stor...	Unresolved	2024-04-11T09:46:27....
9	19	3	The soup is co...	Unresolved	2024-04-11T09:46:27....

```
ComplaintsOnRestaurant:
```



	CID int(10)	OID int(10)
1	14	1
2	15	2
3	16	3
4	19	1

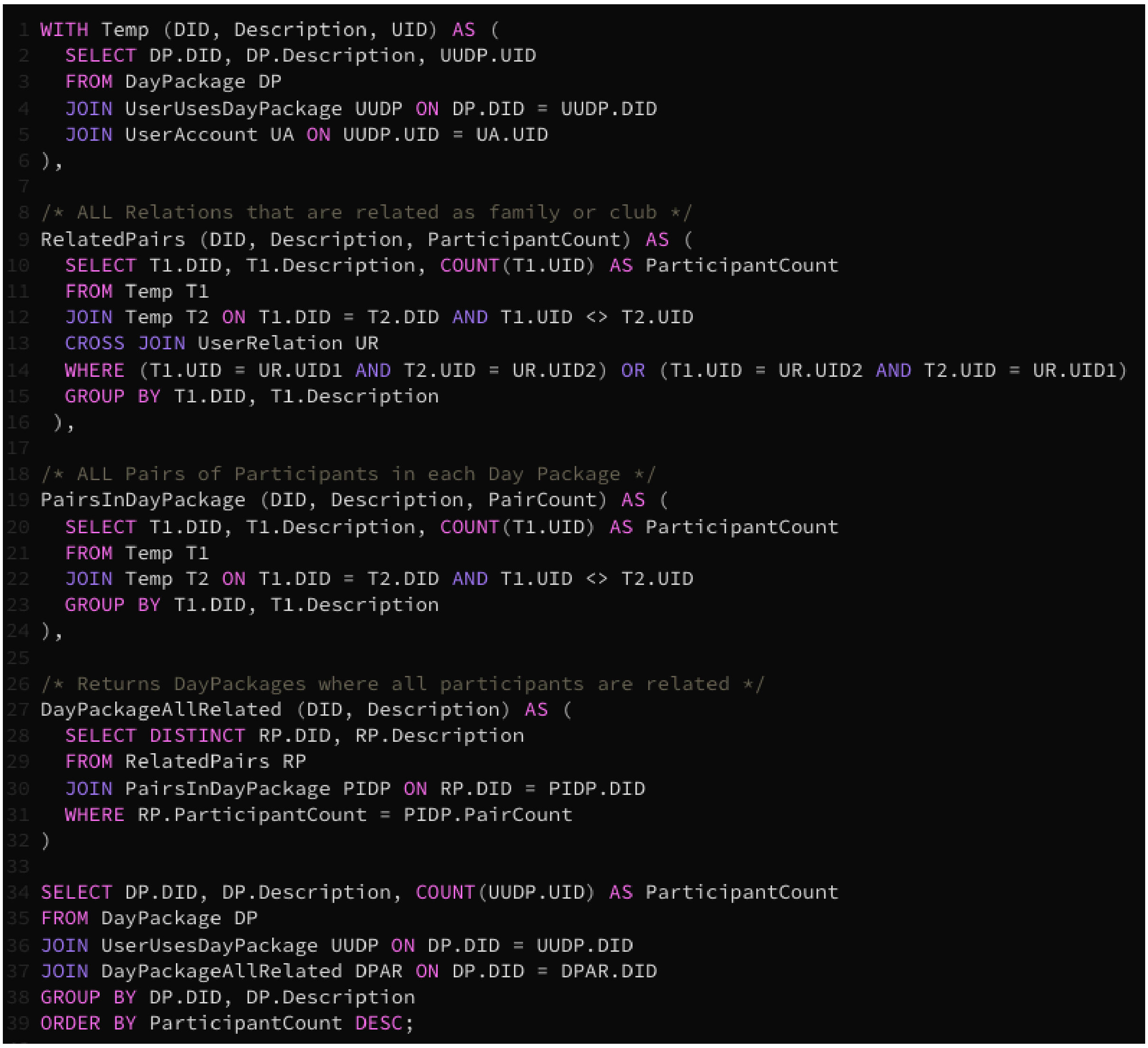
ComplaintsOnShop:



	CID int(10)	SID int(10)
1	11	3
2	12	4
3	13	5
4	17	1
5	18	2

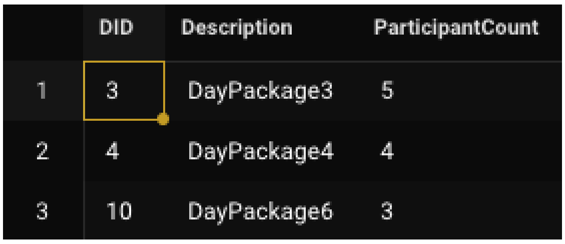
SQL Code For Queries In Appendix B

Query 1 : Find the most popular day packages where all participants are related to one another as either family members or members of the same club.



```sql
WITH Temp (DID, Description, UID) AS (
  SELECT DP.DID, DP.Description, UUDP.UID
  FROM DayPackage DP
  JOIN UserUsesDayPackage UUDP ON DP.DID = UUDP.DID
  JOIN UserAccount UA ON UUDP.UID = UA.UID
),

/* ALL Relations that are related as family or club */
RelatedPairs (DID, Description, ParticipantCount) AS (
  SELECT T1.DID, T1.Description, COUNT(T1.UID) AS ParticipantCount
  FROM Temp T1
  JOIN Temp T2 ON T1.DID = T2.DID AND T1.UID <> T2.UID
  CROSS JOIN UserRelation UR
  WHERE (T1.UID = UR.UID1 AND T2.UID = UR.UID2) OR (T1.UID = UR.UID2 AND T2.UID = UR.UID1)
  GROUP BY T1.DID, T1.Description
 ),

/* ALL Pairs of Participants in each Day Package */
PairsInDayPackage (DID, Description, PairCount) AS (
  SELECT T1.DID, T1.Description, COUNT(T1.UID) AS ParticipantCount
  FROM Temp T1
  JOIN Temp T2 ON T1.DID = T2.DID AND T1.UID <> T2.UID
  GROUP BY T1.DID, T1.Description
),

/* Returns DayPackages where all participants are related */
DayPackageAllRelated (DID, Description) AS (
  SELECT DISTINCT RP.DID, RP.Description
  FROM RelatedPairs RP
  JOIN PairsInDayPackage PIDP ON RP.DID = PIDP.DID
  WHERE RP.ParticipantCount = PIDP.PairCount
)

SELECT DP.DID, DP.Description, COUNT(UUDP.UID) AS ParticipantCount
FROM DayPackage DP
JOIN UserUsesDayPackage UUDP ON DP.DID = UUDP.DID
JOIN DayPackageAllRelated DPAR ON DP.DID = DPAR.DID
GROUP BY DP.DID, DP.Description
ORDER BY ParticipantCount DESC;
```

Output:



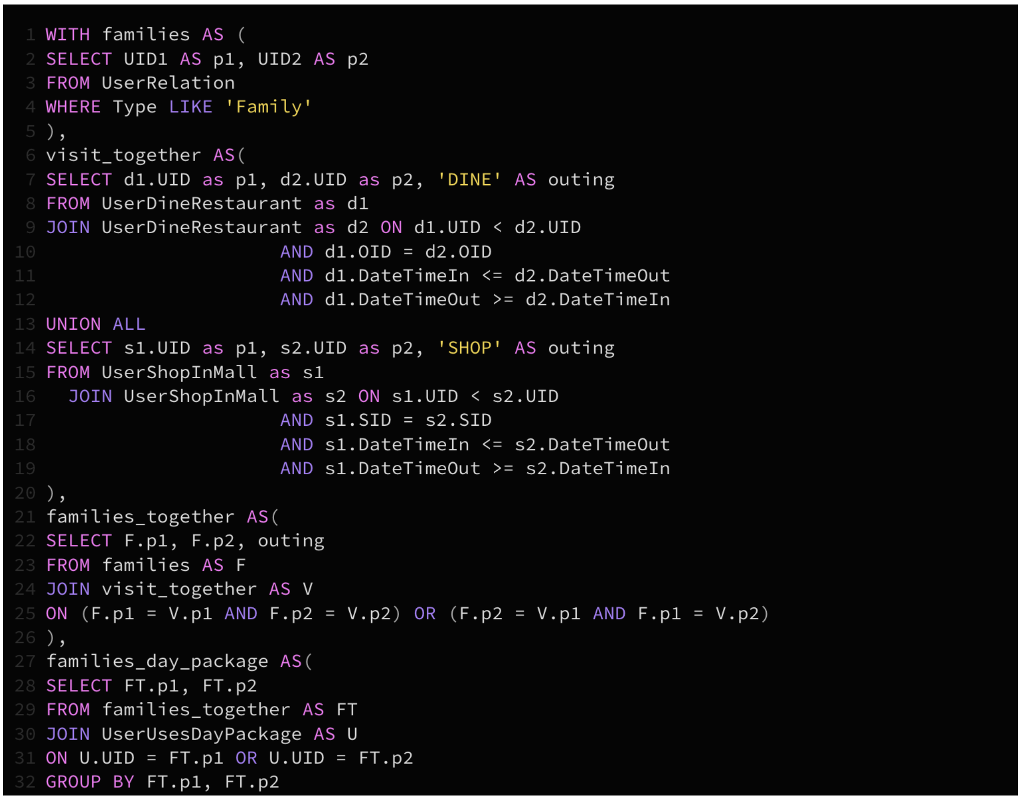
	DID	Description	ParticipantCount
1	3	DayPackage3	5
2	4	DayPackage4	4
3	10	DayPackage6	3

We can tell from the output that the most popular day package where all participants are related is day package 3 with 5 participants, followed by day package 4 with 3 participants and then day package 6 with 3 participants.

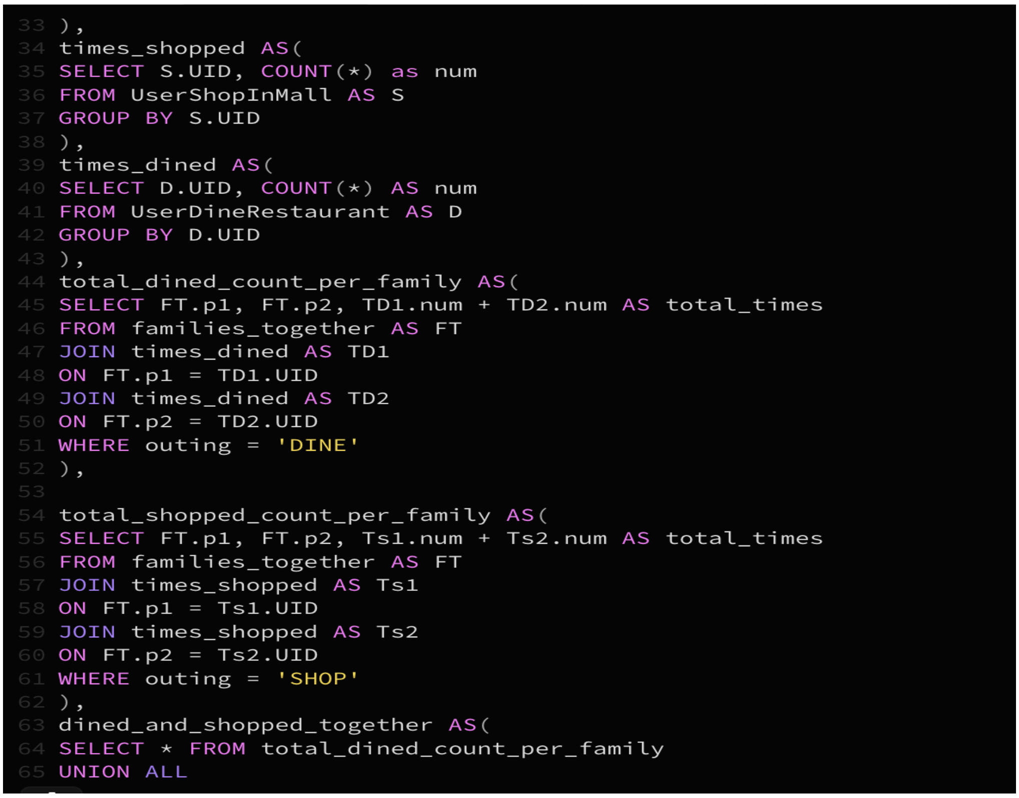
Explanation:

- We want to find the most popular Day Packages where all participants in the Day Package are related to one another as either family members or members of the same club. To do that, we have to utilise the DayPackage table, UserUsesDayPackage table, UserAccount table and the UserRelation table. The UserUsesDayPackage table contains the DID of Day Package and UID of User as entries into the table to show users using the day package, while (UID1, UID2, Type) entries into UserRelation table suggest that User with UID1 have a relationship of either Family or Club member with User with UID2.
- We first created a view Temp that joins DayPackage with UserRelation and UserAccount.
- We then created a view RelatedPairs that gives all related pairs of participants in the respective day packages.
- Another view PairsInDayPackage outputs all the possible number of pairs of participants in the respective day packages, regardless if they are related or not.
- We then compare the results of count in both the RelatedPairs view and PairsInDayPackage view as if for any day package, if the count value of the day package in both views are the same, it would mean that all participants in the day package are related. The result is in the view DayPackageAllRelated.
- Having the DID of all the day packages where all the participants are related, we can join back with DayPackage table and UserUsesDayPackage table to find the number of participants in the day packages where all the participants are related using COUNT()
- The table is arranged in descending order by participant count, suggesting that the most popular day package will be at the top of the output table, which in this case is day package 3.

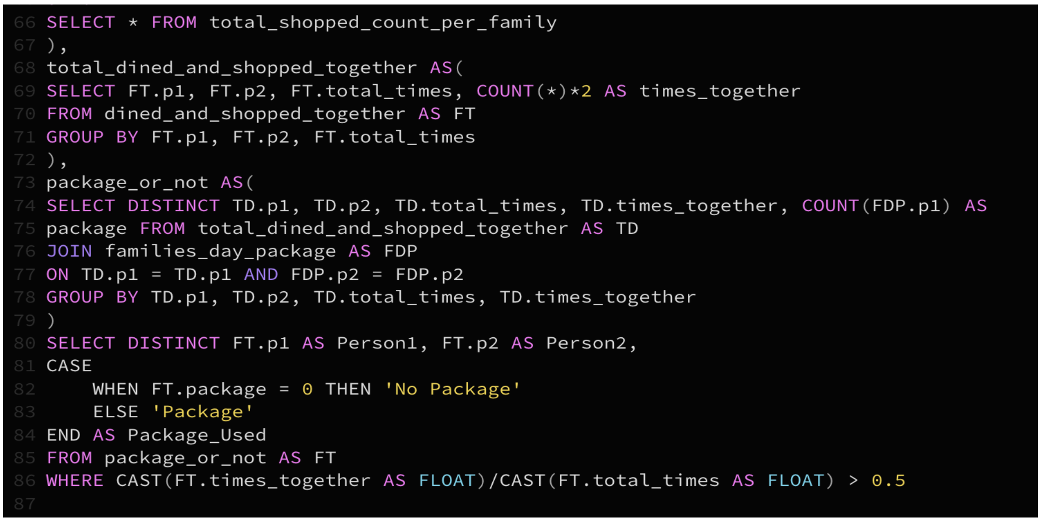
Query 2 : Find families who frequently shopped and dined together, with or without day packages. As part of your output, indicate whether these families use day packages or not. “frequently” means at least 50% of the time.



```
WITH families AS (
SELECT UID1 AS p1, UID2 AS p2
FROM UserRelation
WHERE Type LIKE 'Family'
),
visit_together AS(
SELECT d1.UID as p1, d2.UID as p2, 'DINE' AS outing
FROM UserDineRestaurant as d1
JOIN UserDineRestaurant as d2 ON d1.UID < d2.UID
                     AND d1.OID = d2.OID
                     AND d1.DateTimeIn <= d2.DateTimeOut
                     AND d1.DateTimeOut >= d2.DateTimeIn
UNION ALL
SELECT s1.UID as p1, s2.UID as p2, 'SHOP' AS outing
FROM UserShopInMall as s1
  JOIN UserShopInMall as s2 ON s1.UID < s2.UID
                     AND s1.SID = s2.SID
                     AND s1.DateTimeIn <= s2.DateTimeOut
                     AND s1.DateTimeOut >= s2.DateTimeIn
),
families_together AS(
SELECT F.p1, F.p2, outing
FROM families AS F
JOIN visit_together AS V
ON (F.p1 = V.p1 AND F.p2 = V.p2) OR (F.p2 = V.p1 AND F.p1 = V.p2)
),
families_day_package AS(
SELECT FT.p1, FT.p2
FROM families_together AS FT
JOIN UserUsesDayPackage AS U
ON U.UID = FT.p1 OR U.UID = FT.p2
GROUP BY FT.p1, FT.p2
```

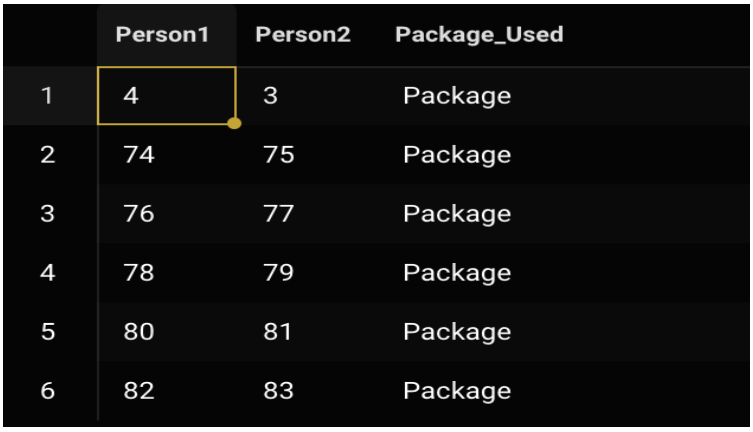


```
),
times_shopped AS(
SELECT S.UID, COUNT(*) as num
FROM UserShopInMall AS S
GROUP BY S.UID
),
times_dined AS(
SELECT D.UID, COUNT(*) AS num
FROM UserDineRestaurant AS D
GROUP BY D.UID
),
total_dined_count_per_family AS(
SELECT FT.p1, FT.p2, TD1.num + TD2.num AS total_times
FROM families_together AS FT
JOIN times_dined AS TD1
ON FT.p1 = TD1.UID
JOIN times_dined AS TD2
ON FT.p2 = TD2.UID
WHERE outing = 'DINE'
),

total_shopped_count_per_family AS(
SELECT FT.p1, FT.p2, Ts1.num + Ts2.num AS total_times
FROM families_together AS FT
JOIN times_shopped AS Ts1
ON FT.p1 = Ts1.UID
JOIN times_shopped AS Ts2
ON FT.p2 = Ts2.UID
WHERE outing = 'SHOP'
),
dined_and_shopped_together AS(
SELECT * FROM total_dined_count_per_family
UNION ALL
```



```
SELECT * FROM total_shopped_count_per_family
),
total_dined_and_shopped_together AS(
SELECT FT.p1, FT.p2, FT.total_times, COUNT(*)*2 AS times_together
FROM dined_and_shopped_together AS FT
GROUP BY FT.p1, FT.p2, FT.total_times
),
package_or_not AS(
SELECT DISTINCT TD.p1, TD.p2, TD.total_times, TD.times_together, COUNT(FDP.p1) AS
package FROM total_dined_and_shopped_together AS TD
JOIN families_day_package AS FDP
ON TD.p1 = TD.p1 AND FDP.p2 = FDP.p2
GROUP BY TD.p1, TD.p2, TD.total_times, TD.times_together
)
SELECT DISTINCT FT.p1 AS Person1, FT.p2 AS Person2,
CASE
    WHEN FT.package = 0 THEN 'No Package'
    ELSE 'Package'
END AS Package_Used
FROM package_or_not AS FT
WHERE CAST(FT.times_together AS FLOAT)/CAST(FT.total_times AS FLOAT) > 0.5
```

Output:



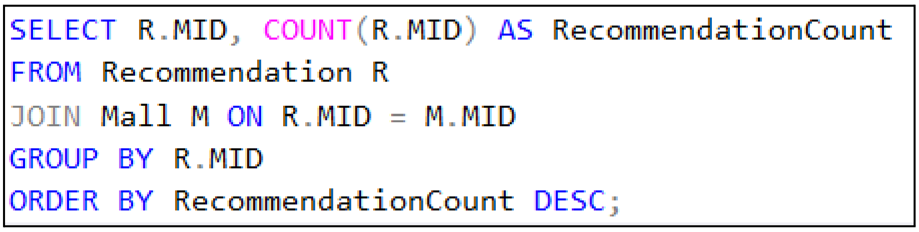
	Person1	Person2	Package_Used
1	4	3	Package
2	74	75	Package
3	76	77	Package
4	78	79	Package
5	80	81	Package
6	82	83	Package

Explanation:

- Identifying Family Relationships: The code starts by selecting pairs of users from a table named "UserRelation" where the relationship type is marked as 'Family'. This step ensures that we're working with family-related user pairs.
- Finding Visits Together: Next, it identifies pairs of users who dined or shopped together based on overlapping visit times. This is done by joining the "UserDineRestaurant" and "UserShopInMall" tables with themselves to find pairs of users who were present at the same place (either a restaurant or a shop) at overlapping time intervals.
- Filtering Family Visits: After identifying visits together, it filters out only those visits where both users belong to the same family, as identified in the first step. This ensures that we're focusing on family-related visits.
- Counting Visits: Then, it counts the number of times each family dined and shopped together separately. This involves counting the occurrences of visits for each family member and summing them up for each family pair.

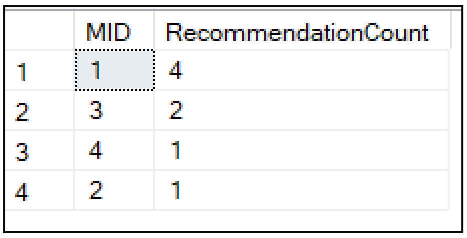
- Determining Day Package Usage: It checks whether each family used day packages by joining the family pairs with the "UserUsesDayPackage" table, which presumably contains information about who has day packages.
- Calculating Usage Ratio:The code calculates the ratio of times a family dined or shopped together to the total number of times they dined or shopped separately.
- Final Output: Finally, the code selects pairs of users where the ratio of times spent together exceeds 50% and indicates whether they used day packages or not.

Query 3: What are the most popular recommendations from the app regarding malls?



```
SELECT R.MID, COUNT(R.MID) AS RecommendationCount
FROM Recommendation R
JOIN Mall M ON R.MID = M.MID
GROUP BY R.MID
ORDER BY RecommendationCount DESC;
```

Output:

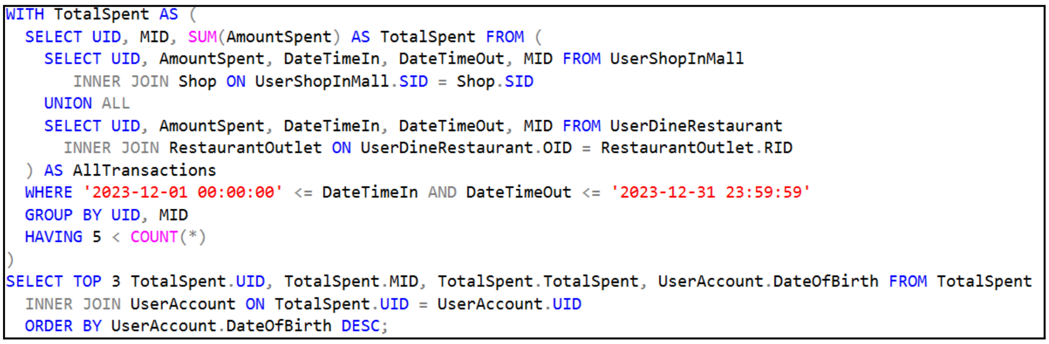


	MID	RecommendationCount
1	1	4
2	3	2
3	4	1
4	2	1

Explanation:

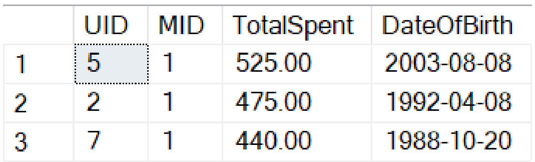
- An inner join between the Recommendation (R) table and the Mall (M) table on the mall ID (MID) is performed to link recommendations made to the corresponding mall.
- The results are grouped by MID to group the recommendation by mall.
- The results are ranked according to the RecommendationCount column in descending order so that the mall with the highest number of recommendations appears first.
- The queries will output the list of MID with their corresponding number of recommendations. The first mall on the list will be the most popular recommendation from the app regarding malls.
- According to the output, the mall with an ID of 1 is the most popular recommendation from the app regarding malls.

Query 4 : Compulsive shoppers are those who have visited a certain mall more than 5 times within a certain period of time. Find the youngest compulsive shoppers and the amount they spent in total during December 2023.

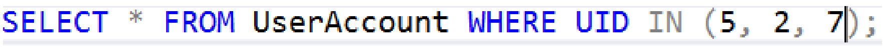


```sql
WITH TotalSpent AS (
  SELECT UID, MID, SUM(AmountSpent) AS TotalSpent FROM (
    SELECT UID, AmountSpent, DateTimeIn, DateTimeOut, MID FROM UserShopInMall
       INNER JOIN Shop ON UserShopInMall.SID = Shop.SID
    UNION ALL
    SELECT UID, AmountSpent, DateTimeIn, DateTimeOut, MID FROM UserDineRestaurant
      INNER JOIN RestaurantOutlet ON UserDineRestaurant.OID = RestaurantOutlet.RID
  ) AS AllTransactions
  WHERE '2023-12-01 00:00:00' <= DateTimeIn AND DateTimeOut <= '2023-12-31 23:59:59'
  GROUP BY UID, MID
  HAVING 5 < COUNT(*)
)
SELECT TOP 3 TotalSpent.UID, TotalSpent.MID, TotalSpent.TotalSpent, UserAccount.DateOfBirth FROM TotalSpent
  INNER JOIN UserAccount ON TotalSpent.UID = UserAccount.UID
  ORDER BY UserAccount.DateOfBirth DESC;
```

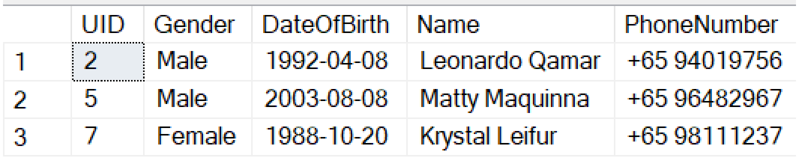
Output:



	UID	MID	TotalSpent	DateOfBirth
1	5	1	525.00	2003-08-08
2	2	1	475.00	1992-04-08
3	7	1	440.00	1988-10-20



```sql
SELECT * FROM UserAccount WHERE UID IN (5, 2, 7);
```



	UID	Gender	DateOfBirth	Name	PhoneNumber
1	2	Male	1992-04-08	Leonardo Qamar	+65 94019756
2	5	Male	2003-08-08	Matty Maquinna	+65 96482967
3	7	Female	1988-10-20	Krystal Leifur	+65 98111237

Explanation:

- We want to find the youngest compulsive shopper who spent the most in a mall that they have visited more than 5 times during December 2023. This query will be able to retrieve this user's UID, MID, their Date of Birth, and total amount spent, by sorting by birthdate.
- Firstly, we combine the user's transactions from both shopping and dining, and calculate the number of visits, as well as the total amount spent.
- Next, we filter the date such that it only gives us the time period of December 2023. We then also filtered the results to only show users who have visited the same mall more than 5 times.
- Lastly, we select the 3 youngest compulsive shoppers by ordering the date of birth in descending order. From there, we can obtain the amount they spent during December 2023.

Query 5 : Find users who have dined in all the restaurants in some malls, but have never dined in any restaurants in some other malls.

```
WITH RestaurantsDinedPerMallPerUser(UID, MID, Count) AS (
   SELECT Dine.UID, MID, COUNT(Distinct Dine.OID) FROM dbo.UserAccount
   INNER JOIN dbo.UserDineRestaurant Dine ON Dine.UID = UserAccount.UID
   INNER JOIN dbo.RestaurantOutlet Restaurant ON Dine.OID = Restaurant.OID
   GROUP BY Dine.UID, Restaurant.MID
), RestaurantsPerMall(MID, Count) AS (
   SELECT RO.MID, COUNT(DISTINCT RO.OID) FROM dbo.Mall
   INNER JOIN dbo.RestaurantOutlet RO on Mall.MID = RO.MID
   GROUP BY RO.MID
), ValidityTable(UID, CountPerUser, Count) AS (
   SELECT UserAccount.UID,RestaurantsDinedPerMallPerUser.Count,
RestaurantsPerMall.Count FROM
   dbo.UserAccount
   INNER JOIN RestaurantsDinedPerMallPerUser ON UserAccount.UID =
RestaurantsDinedPerMallPerUser.UID
   INNER JOIN RestaurantsPerMall ON RestaurantsDinedPerMallPerUser.MID =
RestaurantsPerMall.MID
)
SELECT * FROM
dbo.UserAccount
WHERE UserAccount.UID IN (SELECT UID FROM ValidityTable WHERE
ValidityTable.CountPerUser = ValidityTable.Count)
AND UserAccount.UID NOT IN (SELECT UID FROM ValidityTable WHERE
ValidityTable.CountPerUser <> ValidityTable.Count)
```

Output:

```
+---+------+----------+---------------+--------------+

|UID|Gender|DateOfBirth|Name           |PhoneNumber   |

+---+------+----------+---------------+--------------+

|2  |null  |null       |null           |(736) 124-9365|

|4  |null  |null       |null           |(376) 351-2965|

|7  |null  |null       |null           |(613) 924-8075|

|13 |null  |null       |null           |(365) 948-2170|

|72 |F     |1990-05-15 |Jane Doe       |(943) 508-7261|

|73 |M     |1988-10-20 |Josh Smith     |(267) 894-3501|

|74 |F     |1975-03-08 |Michael Johnson|(510) 627-8439|

+---+------+----------+---------------+--------------+
```

Explanation:

- We count the number of restaurants per mall and also the number of dined restaurants per mall per user.
- We can define that a user is considered to dine at all restaurants in a mall but have no dined at some restaurants in a mall if the following condition holds: there exists a mall such that number of restaurants at the mall = numbers of restaurants dined by the user at the mall AND there exists no mall such that

number of restaurants at the mall != numbers of restaurants dined by the user at the mall.

Query 6 : What are the top 3 highest earning malls and restaurants?

```
SELECT top 3 Revenue, MID, OID
FROM
   (
       SELECT MID, Null as OID, sum(AmountSpent) AS Revenue FROM dbo.Shop
       INNER JOIN dbo.UserShopInMall USIM on Shop.SID = USIM.SID
       GROUP BY MID
       UNION ALL
       SELECT Null as MID, OID, sum(AmountSpent) AS Revenue FROM
dbo.UserDineRestaurant
       GROUP BY OID
   ) AS MallOrRestaurant
ORDER BY Revenue DESC;
```

Output:

```
+---------+----+----+
|Revenue  |MID |OID |
+---------+----+----+
|2295.1600|1   |null|
|1587.4600|null|2   |
|1423.2400|null|3   |
+---------+----+----+
```

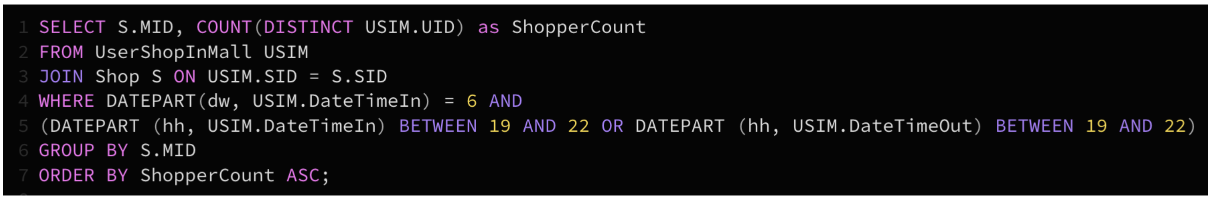
Explanation:

- The revenue of a mall or restaurant is the sum of the amount spent at a mall or restaurant.
- We can union the two summed tables and sort them to find the top 3 revenue.

SQL Code For Additional Queries

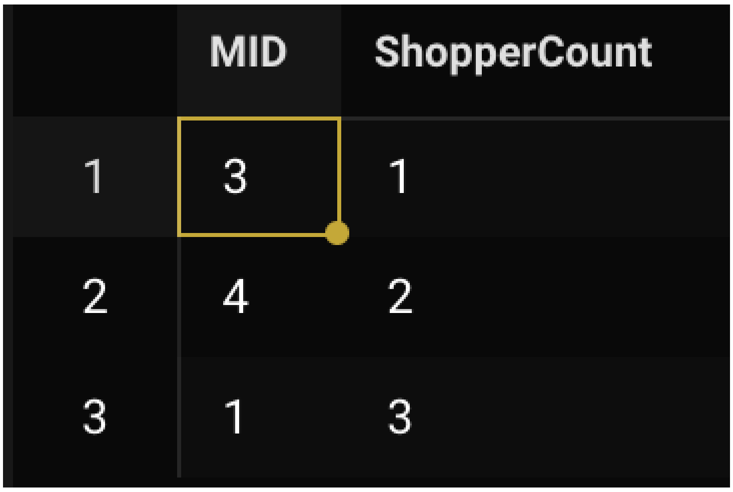
Additional Query: Which malls have the least amount of shoppers on Friday evenings (7pm to 10pm)?

SQL Code:



```sql
SELECT S.MID, COUNT(DISTINCT USIM.UID) as ShopperCount
FROM UserShopInMall USIM
JOIN Shop S ON USIM.SID = S.SID
WHERE DATEPART(dw, USIM.DateTimeIn) = 6 AND
(DATEPART (hh, USIM.DateTimeIn) BETWEEN 19 AND 22 OR DATEPART (hh, USIM.DateTimeOut) BETWEEN 19 AND 22)
GROUP BY S.MID
ORDER BY ShopperCount ASC;
```

Output:



	MID	ShopperCount
1	3	1
2	4	2
3	1	3

Explanation:

- We want to find the malls that have the least shoppers on Friday evenings, so we first join the UserShopInMall table with the Shop table as we want to get the MallID of the shops that the user shops in.
- We then used the DATEPART() function to filter the day of the week and the time
- DATEPART(dw, USIM.DateTimeIn) = 6 filters for dates that are on Friday and DATEPART(hh, USIM.DateTimeIn) and DATEPART(hh, USIM.DateTimeOut) BETWEEN 19 AND 22 filters for time between 7pm to 10pm.
- This would filter out all the shoppers in the respective malls on Friday evenings (7pm to 10pm).
- We then group by the MallID and did a count of distinct shoppers in the mall during that time (Distinct is used as 1 shopper can shop in 2 separate shop in the same mall but should only be counted as 1 shopper)
- To find the mall with the least amount of shoppers, we ordered the table in ascending order and hence the top mall, in this case Mall ID 3, have the least amount of shoppers on Friday evenings.

APPENDIX C: INDIVIDUAL CONTRIBUTION FORM

Individual Contribution form:

Full Name	Individual Contribution to Lab 3 Submission	Percentage of Contribution	Signature
ADRIAN CHUA PIN YU	Create Tables Populate Tables Query 1 Additional Query	20%	
BHUPATHIRAJU MIHIR VARMA	Create Tables Populate Tables Query 2 Additional Query	20%	Mihir
DARRELL MA WEI ZE	Create Tables Populate Tables Query 4 Additional Query	20%	
EUNICE LAI YING XUAN	Create Tables Populate Tables Query 3 Additional Query	20%	Eunice
TAN JING JIE	Create Tables Populate Tables Query 5 & 6 Additional Query	20%	

APPENDIX D: USE OF AI TOOL(S) IN LAB WORK

Each team member should indicate either A or B:

A. I affirm that my contribution(s) to the lab work is my own, produced without help from any AI tool(s).

B. I affirm that my contribution(s) to the lab work has been produced with the use of AI tool(s).

Team member (full name)	Signature	Date	A or B
ADRIAN CHUA PIN YU		11/04/24	A
BHUPATHIRAJU MIHIR VARMA	Mihir	11/04/24	A
DARRELL MA WEI ZE		11/04/24	A
EUNICE LAI YING XUAN	Eunice	11/04/24	A
TAN JING JIE		11/04/24	A

By signing this form, you declare that the above affirmation made is true and that you have read and understood NTU's policy on the use of AI tools.

If any team member answered B, the team member(s) must indicate and replicate the table below for every instance AI tool(s) is used:

Name of AI tool	<For example, ChatGPT>
input prompt	<Insert the question that you asked ChatGPT>
Date generated	
Output generated	<Insert the response verbatim from ChatGPT>
Output screenshots	
Impact on submission	<Briefly explain which part of your submitted work was ChatGPT's response applied>