CZ2007:

Introduction to Databases

Lab 5

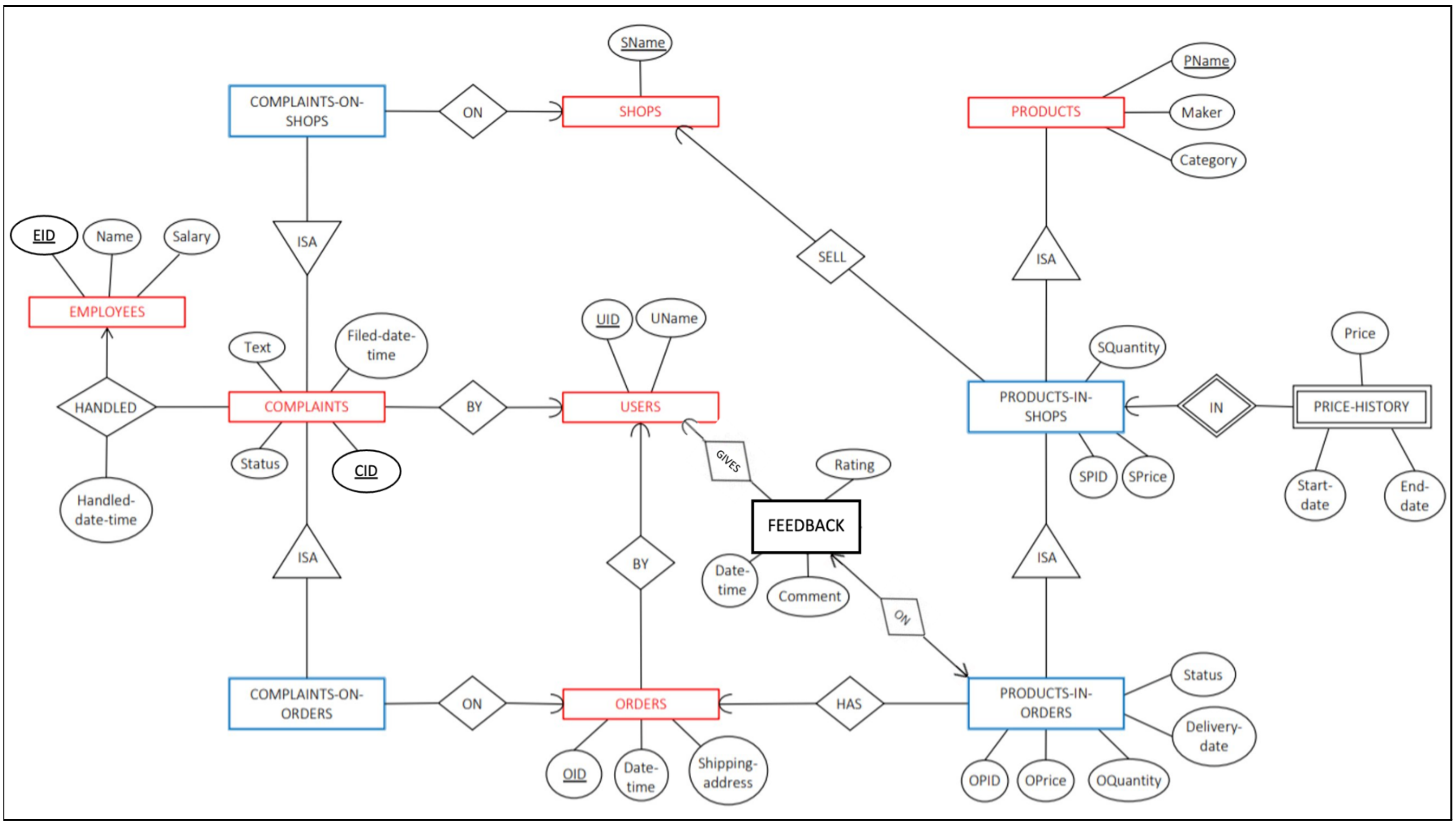
Implementation of Database and SQL Queries

SS4 – Group 3

Name	Matric Number
Dhanyamraju Harsh Rao	U2023045C
Jaiswal Arnav	U2023834A
Arora Kanupriya	U2023174C
Neel Kumar	U2023314J
Malavade Sanskar Deepak	U2023184A
Parashar Kshitij	U2023805J

Table of Contents

ER Diagram



SName
PName
COMPLAINTS-ON-SHOPS
ON
SHOPS
PRODUCTS
Maker
Category
EID
Name
Salary
ISA
SELL
ISA
EMPLOYEES
UID
UName
Text
Filed-date-time
SQuantity
Price
HANDLED
COMPLAINTS
BY
USERS
PRODUCTS-IN-SHOPS
IN
PRICE-HISTORY
Status
CID
GIVES
Rating
SPID
SPrice
Start-date
End-date
Handled-date-time
FEEDBACK
ISA
BY
Date-time
Comment
ISA
ON
Status
COMPLAINTS-ON-ORDERS
ON
ORDERS
HAS
PRODUCTS-IN-ORDERS
Delivery-date
OID
Date-time
Shipping-address
OPID
OPrice
OQuantity

Table Creation SQL DDL Commands

Products

```
 CREATE TABLE PRODUCTS (
      PName VARCHAR (50) NOT NULL,
      Maker VARCHAR (50),
      Category VARCHAR (50),
      PRIMARY KEY (Pname),
);
```

Products In Shops

```
CREATE TABLE PRODUCTS_IN_SHOPS (
      PName VARCHAR (50) NOT NULL,
      SName VARCHAR (50) NOT NULL,
      SQuantity INT NOT NULL CHECK (SQuantity >0),
      SPID INT,
      SPrice DECIMAL(10, 2) NOT NULL CHECK (SPrice >0),
      PRIMARY KEY (PName, SName),
      foreign key (PName) references PRODUCTS(PName)
      ON DELETE CASCADE ON UPDATE CASCADE,
      foreign key (SName) references SHOPS(SName)
      ON DELETE CASCADE ON UPDATE CASCADE,

);
```

Note: refer to additional efforts at the end

Products In Orders

```
CREATE TABLE PRODUCTS_IN_ORDER (
	PName VARCHAR (50) NOT NULL,
	SName VARCHAR (50) NOT NULL,
	OID VARCHAR (20) NOT NULL,
	OQuantity INT NOT NULL CHECK (OQuantity >0),
	OPID INT,
	OPrice DECIMAL(10, 2) NOT NULL CHECK (OPrice >0),
	OrderStatus INT NOT NULL,
	DeliveryDate smalldatetime,
	PRIMARY KEY (PName, SName, OID),
	foreign key (PName, SName) references
PRODUCTS_IN_SHOPS(PName, SName)
	ON DELETE CASCADE ON UPDATE CASCADE,
	foreign key (OID) references ORDERS(OID)
	ON DELETE CASCADE ON UPDATE CASCADE,
);
```

`Note:` refer to additional efforts at the end

Price History

```
CREATE TABLE PRICE_HISTORY (
	PName VARCHAR (50) NOT NULL,
	SName VARCHAR (50) NOT NULL,
	Price INT NOT NULL CHECK (Price >0),
	Start_Date smalldatetime NOT NULL,
	End_Date smalldatetime NOT NULL,
	PRIMARY KEY (PName, SName, Start_Date),
	foreign key (PName, SName) references PRODUCTS_IN_SHOPS(PName,
SName)
	ON DELETE CASCADE ON UPDATE CASCADE,
	CHECK (Start_Date < End_Date)
);
```

`Note:` refer to additional efforts at the end

Feedback

```
CREATE TABLE Feedback (
           PName VARCHAR(50) NOT NULL,
           SName VARCHAR(50) NOT NULL,
           OID VARCHAR(20) NOT NULL,
           Date DATE NOT NULL,
           UID VARCHAR(20) NOT NULL,
           Ratings TINYINT NOT NULL CHECK (Ratings>0 AND Ratings<=5
),
           Comments VARCHAR(MAX) NULL,
           PRIMARY KEY (PName, SName, OID),
           foreign key (PName, SName, OID) references
PRODUCTS_IN_ORDER(PName, SName, OID)
           ON DELETE CASCADE ON UPDATE CASCADE,
           foreign key (UID) references USERS(UID)
           ON DELETE NO ACTION ON UPDATE NO ACTION,
           );
```

Shops

```
 CREATE TABLE SHOPS (
       SNAME VARCHAR (50) NOT NULL,
       PRIMARY KEY (SNAME),
 );
```

Orders

```
 CREATE TABLE ORDERS (
       OID VARCHAR(20) PRIMARY KEY NOT NULL,
       Date_Time SMALLDATETIME NOT NULL,
       Shipping_Address VARCHAR(50) NOT NULL,
       UID VARCHAR(20) FOREIGN KEY REFERENCES USERS(UID)
       ON DELETE CASCADE ON UPDATE CASCADE,
 );
```

Users

```
CREATE TABLE USERS (
      UID VARCHAR(20) NOT NULL,
      UNAME VARCHAR(30) NOT NULL,
      PRIMARY KEY (UID),
);
```

Employee

```
CREATE TABLE EMPLOYEE (
  EID VARCHAR(20) NOT NULL,
  ENAME VARCHAR(30) NOT NULL,
  SALARY INT CHECK (SALARY >0),
  PRIMARY KEY (EID),
);
```

Complaints

```
CREATE TABLE COMPLAINTS(
   CID varchar(20) NOT NULL,
   COMPLAINT_TEXT varchar(200) NULL,
   FILED_DATE_TIME datetime NULL,
   HANDLED_DATE_TIME datetime NULL,
   COMPLAINT_STATUS varchar(50) NULL,
   EID varchar(50) NULL,
   UID varchar(20) NULL,
   PRIMARY KEY (CID),
   foreign key (EID) references EMPLOYEE(EID)
           ON DELETE CASCADE ON UPDATE CASCADE,
   foreign key (UID) references USERS(UID)
           ON DELETE CASCADE ON UPDATE CASCADE,
  CHECK (FILED_DATE_TIME < HANDLED_DATE_TIME)
);
```

Complaints on Shops

```
CREATE TABLE COMPLAINTS_ON_SHOP (
           SName VARCHAR (50) NULL,
           CID VARCHAR(20) NOT NULL,
           PRIMARY KEY (CID),
           foreign key (SName) references SHOPS(SName)
           ON DELETE CASCADE ON UPDATE CASCADE,
           foreign key (CID) references COMPLAINTS(CID)
           ON DELETE CASCADE ON UPDATE CASCADE,
);
```

Complaints on Orders

```
CREATE TABLE COMPLAINTS_ON_ORDER (
           OID VARCHAR (20) NULL,
           CID VARCHAR(20) NOT NULL,
           PRIMARY KEY (CID),
           foreign key (OID) references ORDERS(OID)
           ON DELETE CASCADE ON UPDATE CASCADE,
           foreign key (CID) references COMPLAINTS(CID)
           ON DELETE CASCADE ON UPDATE CASCADE,
);
```

Database Population

All the data for our tables were first created using python to establish and maintain dependencies amongst data variables. After doing so, we scripted the data into the SQL server using the command mentioned below for all the tables. (https://sqlizer.io/#/)

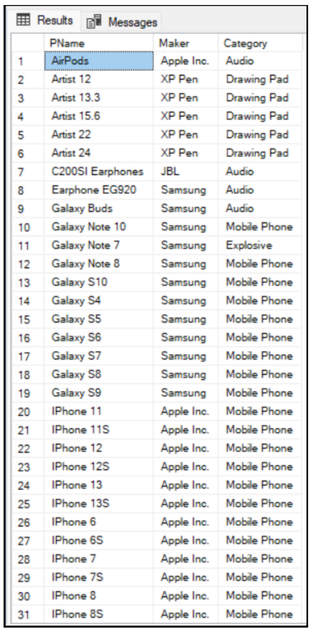
Here is an example for one such table, Complaints. The format for entering the data for each record here is (CID, Complaint_Text, Filed_Date_Time, Handled_Date_Time, Status, EID, UID)

```
DELETE FROM COMPLAINTS;
GO
INSERT INTO COMPLAINTS VALUES
('C0001','Product packaging was bad','2021-05-21
13:32:00',NULL,'Pending','E0001','U308'),
('C0004','The shipping price is too high','2021-05-25
22:10:00',NULL,'Pending','E0001','U466'),
('C0007','I did not receive order on time','2021-05-21
13:32:00',NULL,'Pending','E0001','U197'),
('C0010','I didn't like the way the seller chatted with
me','2021-08-01 19:32:00',NULL,'Pending','E0001','U416'),
..//for all the data values
```

Table Records

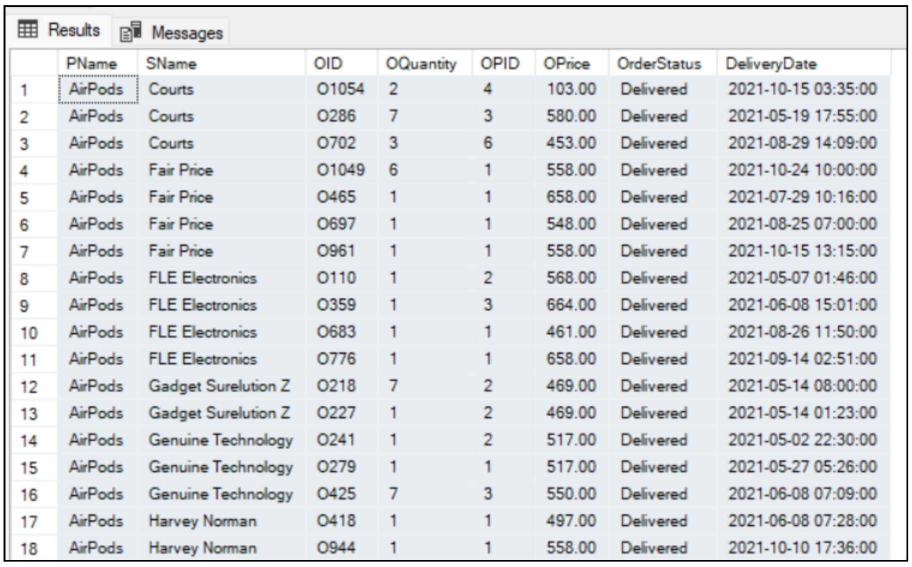
Since our database size was quite large for most of our tables (some exceeding 3500 records since we used python scripts to match all dependencies where necessary), it was not feasible to attach the entire printout in this report. Thus, we printed out all the records of the outputs to an excel workbook where all the different tables are entered as separate sheets. **The excel file is attached in the zip folder** alongside the MP4 recordings, and screenshots of some of the entries in our tables is attached below for your perusal.

Products

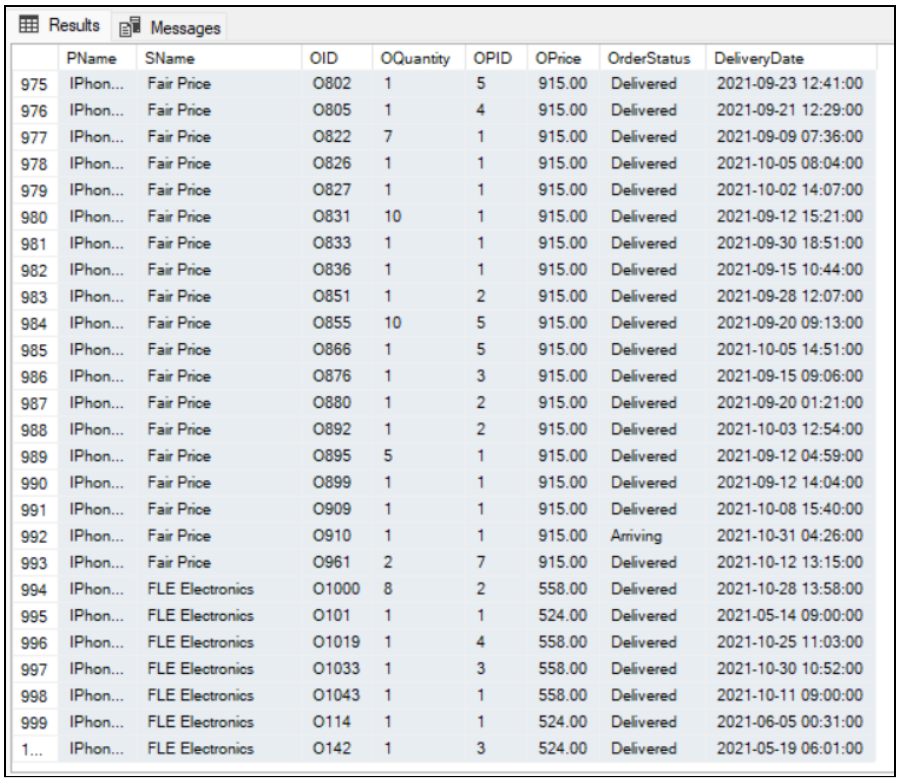


	PName	Maker	Category
1	AirPods	Apple Inc.	Audio
2	Artist 12	XP Pen	Drawing Pad
3	Artist 13.3	XP Pen	Drawing Pad
4	Artist 15.6	XP Pen	Drawing Pad
5	Artist 22	XP Pen	Drawing Pad
6	Artist 24	XP Pen	Drawing Pad
7	C200SI Earphones	JBL	Audio
8	Earphone EG920	Samsung	Audio
9	Galaxy Buds	Samsung	Audio
10	Galaxy Note 10	Samsung	Mobile Phone
11	Galaxy Note 7	Samsung	Explosive
12	Galaxy Note 8	Samsung	Mobile Phone
13	Galaxy S10	Samsung	Mobile Phone
14	Galaxy S4	Samsung	Mobile Phone
15	Galaxy S5	Samsung	Mobile Phone
16	Galaxy S6	Samsung	Mobile Phone
17	Galaxy S7	Samsung	Mobile Phone
18	Galaxy S8	Samsung	Mobile Phone
19	Galaxy S9	Samsung	Mobile Phone
20	IPhone 11	Apple Inc.	Mobile Phone
21	IPhone 11S	Apple Inc.	Mobile Phone
22	IPhone 12	Apple Inc.	Mobile Phone
23	IPhone 12S	Apple Inc.	Mobile Phone
24	IPhone 13	Apple Inc.	Mobile Phone
25	IPhone 13S	Apple Inc.	Mobile Phone
26	IPhone 6	Apple Inc.	Mobile Phone
27	IPhone 6S	Apple Inc.	Mobile Phone
28	IPhone 7	Apple Inc.	Mobile Phone
29	IPhone 7S	Apple Inc.	Mobile Phone
30	IPhone 8	Apple Inc.	Mobile Phone
31	IPhone 8S	Apple Inc.	Mobile Phone

Products In Order

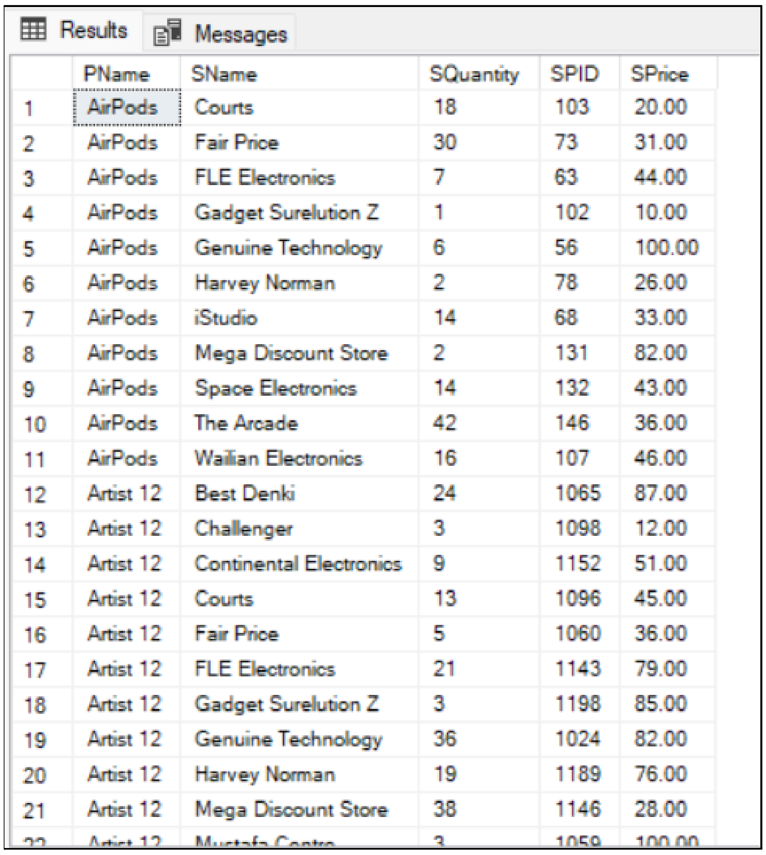


	PName	SName	OID	OQuantity	OPID	OPrice	OrderStatus	DeliveryDate
1	AirPods	Courts	O1054	2	4	103.00	Delivered	2021-10-15 03:35:00
2	AirPods	Courts	O286	7	3	580.00	Delivered	2021-05-19 17:55:00
3	AirPods	Courts	O702	3	6	453.00	Delivered	2021-08-29 14:09:00
4	AirPods	Fair Price	O1049	6	1	558.00	Delivered	2021-10-24 10:00:00
5	AirPods	Fair Price	O465	1	1	658.00	Delivered	2021-07-29 10:16:00
6	AirPods	Fair Price	O697	1	1	548.00	Delivered	2021-08-25 07:00:00
7	AirPods	Fair Price	O961	1	1	558.00	Delivered	2021-10-15 13:15:00
8	AirPods	FLE Electronics	O110	1	2	568.00	Delivered	2021-05-07 01:46:00
9	AirPods	FLE Electronics	O359	1	3	664.00	Delivered	2021-06-08 15:01:00
10	AirPods	FLE Electronics	O683	1	1	461.00	Delivered	2021-08-26 11:50:00
11	AirPods	FLE Electronics	O776	1	1	658.00	Delivered	2021-09-14 02:51:00
12	AirPods	Gadget Surelution Z	O218	7	2	469.00	Delivered	2021-05-14 08:00:00
13	AirPods	Gadget Surelution Z	O227	1	2	469.00	Delivered	2021-05-14 01:23:00
14	AirPods	Genuine Technology	O241	1	2	517.00	Delivered	2021-05-02 22:30:00
15	AirPods	Genuine Technology	O279	1	1	517.00	Delivered	2021-05-27 05:26:00
16	AirPods	Genuine Technology	O425	7	3	550.00	Delivered	2021-06-08 07:09:00
17	AirPods	Harvey Norman	O418	1	1	497.00	Delivered	2021-06-08 07:28:00
18	AirPods	Harvey Norman	O944	1	1	558.00	Delivered	2021-10-10 17:36:00



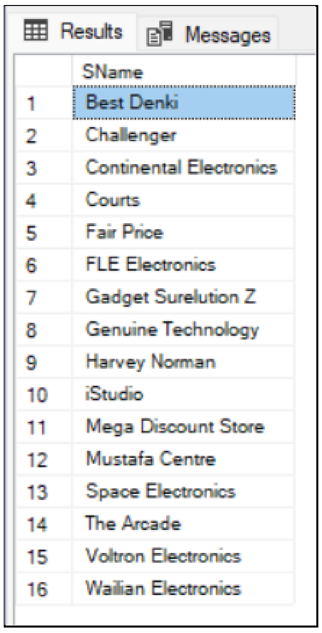
	PName	SName	OID	OQuantity	OPID	OPrice	OrderStatus	DeliveryDate
975	IPhon...	Fair Price	O802	1	5	915.00	Delivered	2021-09-23 12:41:00
976	IPhon...	Fair Price	O805	1	4	915.00	Delivered	2021-09-21 12:29:00
977	IPhon...	Fair Price	O822	7	1	915.00	Delivered	2021-09-09 07:36:00
978	IPhon...	Fair Price	O826	1	1	915.00	Delivered	2021-10-05 08:04:00
979	IPhon...	Fair Price	O827	1	1	915.00	Delivered	2021-10-02 14:07:00
980	IPhon...	Fair Price	O831	10	1	915.00	Delivered	2021-09-12 15:21:00
981	IPhon...	Fair Price	O833	1	1	915.00	Delivered	2021-09-30 18:51:00
982	IPhon...	Fair Price	O836	1	1	915.00	Delivered	2021-09-15 10:44:00
983	IPhon...	Fair Price	O851	1	2	915.00	Delivered	2021-09-28 12:07:00
984	IPhon...	Fair Price	O855	10	5	915.00	Delivered	2021-09-20 09:13:00
985	IPhon...	Fair Price	O866	1	5	915.00	Delivered	2021-10-05 14:51:00
986	IPhon...	Fair Price	O876	1	3	915.00	Delivered	2021-09-15 09:06:00
987	IPhon...	Fair Price	O880	1	2	915.00	Delivered	2021-09-20 01:21:00
988	IPhon...	Fair Price	O892	1	2	915.00	Delivered	2021-10-03 12:54:00
989	IPhon...	Fair Price	O895	5	1	915.00	Delivered	2021-09-12 04:59:00
990	IPhon...	Fair Price	O899	1	1	915.00	Delivered	2021-09-12 14:04:00
991	IPhon...	Fair Price	O909	1	1	915.00	Delivered	2021-10-08 15:40:00
992	IPhon...	Fair Price	O910	1	1	915.00	Arriving	2021-10-31 04:26:00
993	IPhon...	Fair Price	O961	2	7	915.00	Delivered	2021-10-12 13:15:00
994	IPhon...	FLE Electronics	O1000	8	2	558.00	Delivered	2021-10-28 13:58:00
995	IPhon...	FLE Electronics	O101	1	1	524.00	Delivered	2021-05-14 09:00:00
996	IPhon...	FLE Electronics	O1019	1	4	558.00	Delivered	2021-10-25 11:03:00
997	IPhon...	FLE Electronics	O1033	1	3	558.00	Delivered	2021-10-30 10:52:00
998	IPhon...	FLE Electronics	O1043	1	1	558.00	Delivered	2021-10-11 09:00:00
999	IPhon...	FLE Electronics	O114	1	1	524.00	Delivered	2021-06-05 00:31:00
1...	IPhon...	FLE Electronics	O142	1	3	524.00	Delivered	2021-05-19 06:01:00

Products in Shops



	PName	SName	SQuantity	SPID	SPrice
1	AirPods	Courts	18	103	20.00
2	AirPods	Fair Price	30	73	31.00
3	AirPods	FLE Electronics	7	63	44.00
4	AirPods	Gadget Surelution Z	1	102	10.00
5	AirPods	Genuine Technology	6	56	100.00
6	AirPods	Harvey Norman	2	78	26.00
7	AirPods	iStudio	14	68	33.00
8	AirPods	Mega Discount Store	2	131	82.00
9	AirPods	Space Electronics	14	132	43.00
10	AirPods	The Arcade	42	146	36.00
11	AirPods	Wailian Electronics	16	107	46.00
12	Artist 12	Best Denki	24	1065	87.00
13	Artist 12	Challenger	3	1098	12.00
14	Artist 12	Continental Electronics	9	1152	51.00
15	Artist 12	Courts	13	1096	45.00
16	Artist 12	Fair Price	5	1060	36.00
17	Artist 12	FLE Electronics	21	1143	79.00
18	Artist 12	Gadget Surelution Z	3	1198	85.00
19	Artist 12	Genuine Technology	36	1024	82.00
20	Artist 12	Harvey Norman	19	1189	76.00
21	Artist 12	Mega Discount Store	38	1146	28.00
22	Artist 12	Mustafa Centre	3	1059	100.00

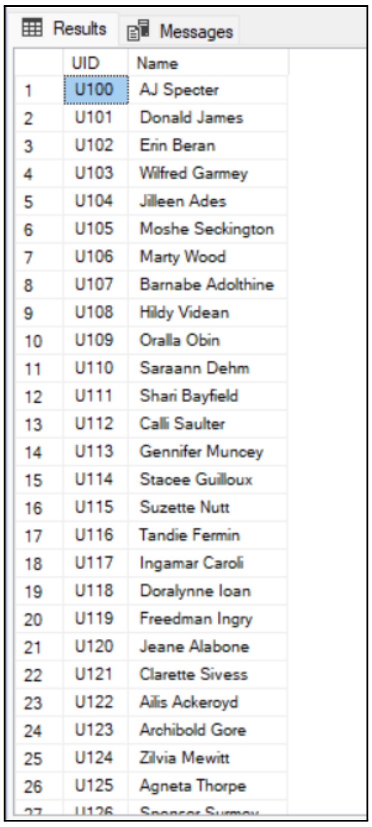
Shops



Results | Messages

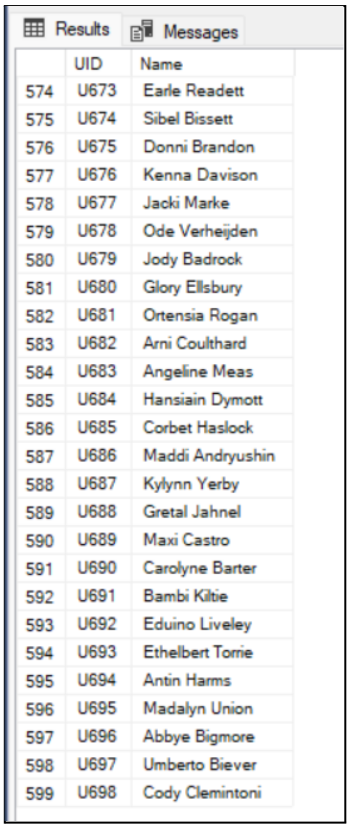
	SName
1	Best Denki
2	Challenger
3	Continental Electronics
4	Courts
5	Fair Price
6	FLE Electronics
7	Gadget Surelution Z
8	Genuine Technology
9	Harvey Norman
10	iStudio
11	Mega Discount Store
12	Mustafa Centre
13	Space Electronics
14	The Arcade
15	Voltron Electronics
16	Wailian Electronics

Users



Results | Messages

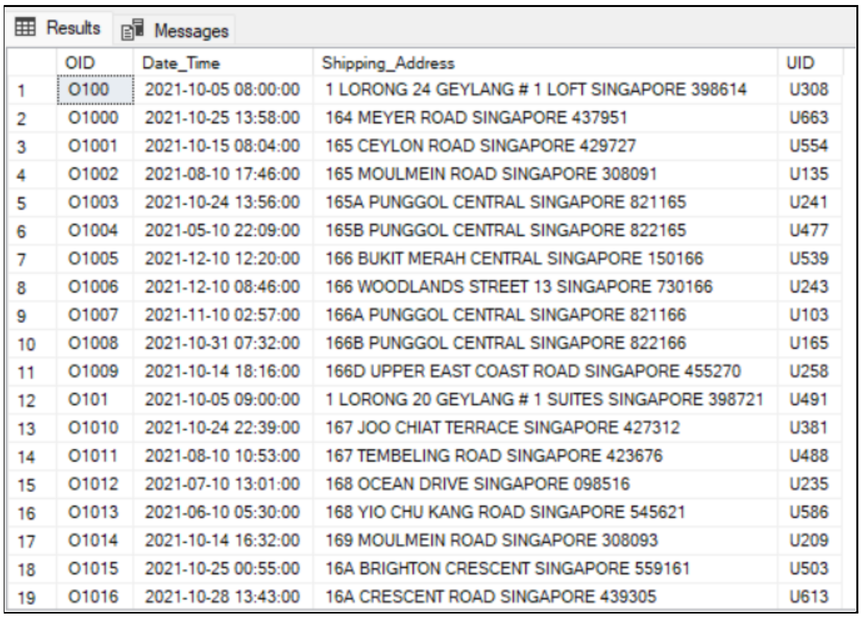
	UID	Name
1	U100	AJ Specter
2	U101	Donald James
3	U102	Erin Beran
4	U103	Wilfred Garney
5	U104	Jilleen Ades
6	U105	Moshe Seckington
7	U106	Marty Wood
8	U107	Barnabe Adolthine
9	U108	Hildy Videan
10	U109	Oralla Obin
11	U110	Saraann Dehm
12	U111	Shari Bayfield
13	U112	Calli Saulter
14	U113	Gennifer Muncey
15	U114	Stacee Guilloux
16	U115	Suzette Nutt
17	U116	Tandie Fermin
18	U117	Ingamar Caroli
19	U118	Doralynne Ioan
20	U119	Freedman Ingry
21	U120	Jeane Alabone
22	U121	Clarette Sivess
23	U122	Ailis Ackeroyd
24	U123	Archibold Gore
25	U124	Zilvia Mewitt
26	U125	Agneta Thorpe
27	U126	Spencer Surmey



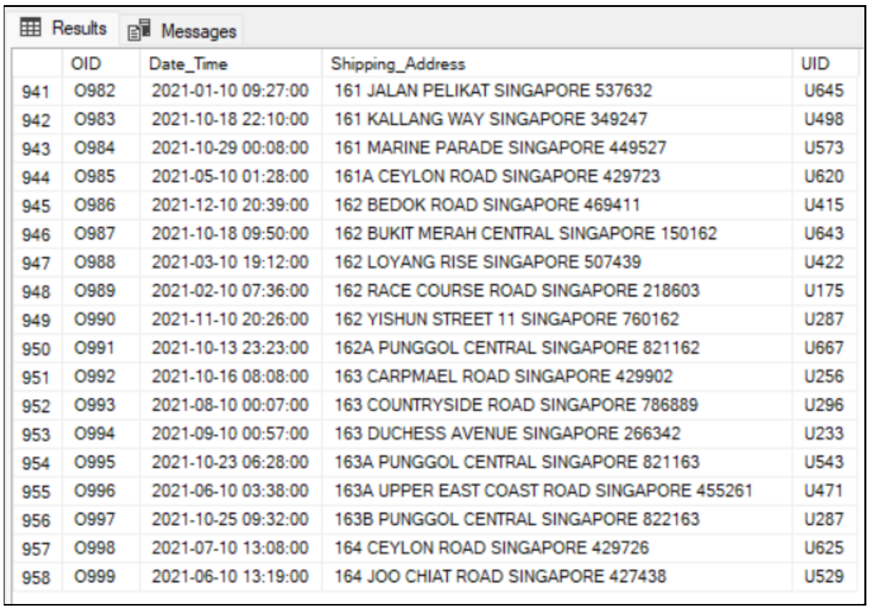
Results | Messages

	UID	Name
574	U673	Earle Readett
575	U674	Sibel Bissett
576	U675	Donni Brandon
577	U676	Kenna Davison
578	U677	Jacki Marke
579	U678	Ode Verheijden
580	U679	Jody Badrock
581	U680	Glory Ellsbury
582	U681	Ortensia Rogan
583	U682	Arni Coulthard
584	U683	Angeline Meas
585	U684	Hansiain Dymott
586	U685	Corbet Haslock
587	U686	Maddi Andryushin
588	U687	Kylynn Yerby
589	U688	Gretal Jahnel
590	U689	Maxi Castro
591	U690	Carolyne Barter
592	U691	Bambi Kiltie
593	U692	Eduino Liveley
594	U693	Ethelbert Torrie
595	U694	Antin Harms
596	U695	Madalyn Union
597	U696	Abbye Bigmore
598	U697	Umberto Biever
599	U698	Cody Clemintoni

Orders

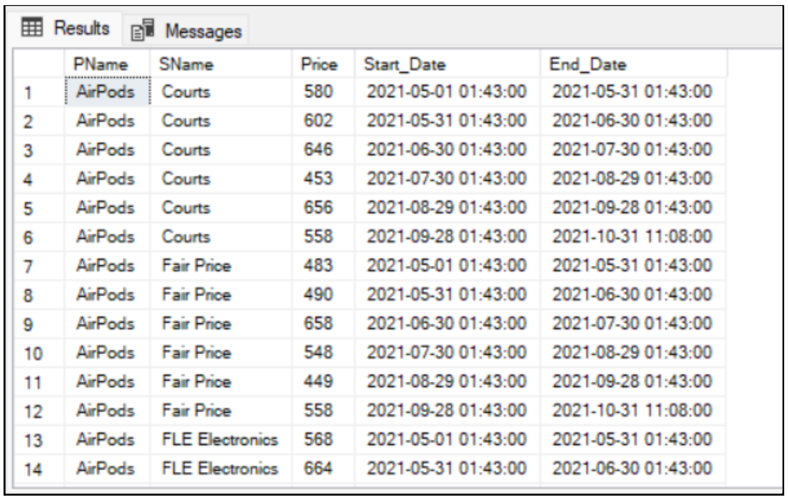


	OID	Date_Time	Shipping_Address	UID
1	O100	2021-10-05 08:00:00	1 LORONG 24 GEYLANG # 1 LOFT SINGAPORE 398614	U308
2	O1000	2021-10-25 13:58:00	164 MEYER ROAD SINGAPORE 437951	U663
3	O1001	2021-10-15 08:04:00	165 CEYLON ROAD SINGAPORE 429727	U554
4	O1002	2021-08-10 17:46:00	165 MOULMEIN ROAD SINGAPORE 308091	U135
5	O1003	2021-10-24 13:56:00	165A PUNGGOL CENTRAL SINGAPORE 821165	U241
6	O1004	2021-05-10 22:09:00	165B PUNGGOL CENTRAL SINGAPORE 822165	U477
7	O1005	2021-12-10 12:20:00	166 BUKIT MERAH CENTRAL SINGAPORE 150166	U539
8	O1006	2021-12-10 08:46:00	166 WOODLANDS STREET 13 SINGAPORE 730166	U243
9	O1007	2021-11-10 02:57:00	166A PUNGGOL CENTRAL SINGAPORE 821166	U103
10	O1008	2021-10-31 07:32:00	166B PUNGGOL CENTRAL SINGAPORE 822166	U165
11	O1009	2021-10-14 18:16:00	166D UPPER EAST COAST ROAD SINGAPORE 455270	U258
12	O101	2021-10-05 09:00:00	1 LORONG 20 GEYLANG # 1 SUITES SINGAPORE 398721	U491
13	O1010	2021-10-24 22:39:00	167 JOO CHIAT TERRACE SINGAPORE 427312	U381
14	O1011	2021-08-10 10:53:00	167 TEMBELING ROAD SINGAPORE 423676	U488
15	O1012	2021-07-10 13:01:00	168 OCEAN DRIVE SINGAPORE 098516	U235
16	O1013	2021-06-10 05:30:00	168 YIO CHU KANG ROAD SINGAPORE 545621	U586
17	O1014	2021-10-14 16:32:00	169 MOULMEIN ROAD SINGAPORE 308093	U209
18	O1015	2021-10-25 00:55:00	16A BRIGHTON CRESCENT SINGAPORE 559161	U503
19	O1016	2021-10-28 13:43:00	16A CRESCENT ROAD SINGAPORE 439305	U613

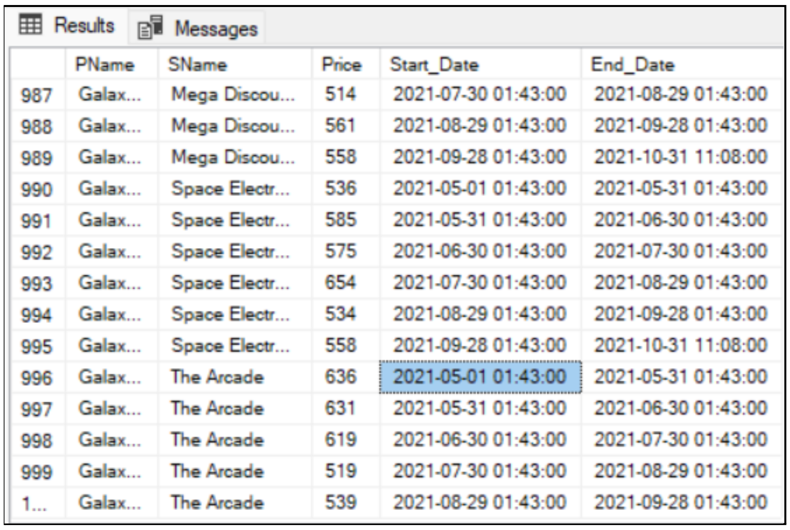


	OID	Date_Time	Shipping_Address	UID
941	O982	2021-01-10 09:27:00	161 JALAN PELIKAT SINGAPORE 537632	U645
942	O983	2021-10-18 22:10:00	161 KALLANG WAY SINGAPORE 349247	U498
943	O984	2021-10-29 00:08:00	161 MARINE PARADE SINGAPORE 449527	U573
944	O985	2021-05-10 01:28:00	161A CEYLON ROAD SINGAPORE 429723	U620
945	O986	2021-12-10 20:39:00	162 BEDOK ROAD SINGAPORE 469411	U415
946	O987	2021-10-18 09:50:00	162 BUKIT MERAH CENTRAL SINGAPORE 150162	U643
947	O988	2021-03-10 19:12:00	162 LOYANG RISE SINGAPORE 507439	U422
948	O989	2021-02-10 07:36:00	162 RACE COURSE ROAD SINGAPORE 218603	U175
949	O990	2021-11-10 20:26:00	162 YISHUN STREET 11 SINGAPORE 760162	U287
950	O991	2021-10-13 23:23:00	162A PUNGGOL CENTRAL SINGAPORE 821162	U667
951	O992	2021-10-16 08:08:00	163 CARPMAEL ROAD SINGAPORE 429902	U256
952	O993	2021-08-10 00:07:00	163 COUNTRYSIDE ROAD SINGAPORE 786889	U296
953	O994	2021-09-10 00:57:00	163 DUCHESS AVENUE SINGAPORE 266342	U233
954	O995	2021-10-23 06:28:00	163A PUNGGOL CENTRAL SINGAPORE 821163	U543
955	O996	2021-06-10 03:38:00	163A UPPER EAST COAST ROAD SINGAPORE 455261	U471
956	O997	2021-10-25 09:32:00	163B PUNGGOL CENTRAL SINGAPORE 822163	U287
957	O998	2021-07-10 13:08:00	164 CEYLON ROAD SINGAPORE 429726	U625
958	O999	2021-06-10 13:19:00	164 JOO CHIAT ROAD SINGAPORE 427438	U529

Price History

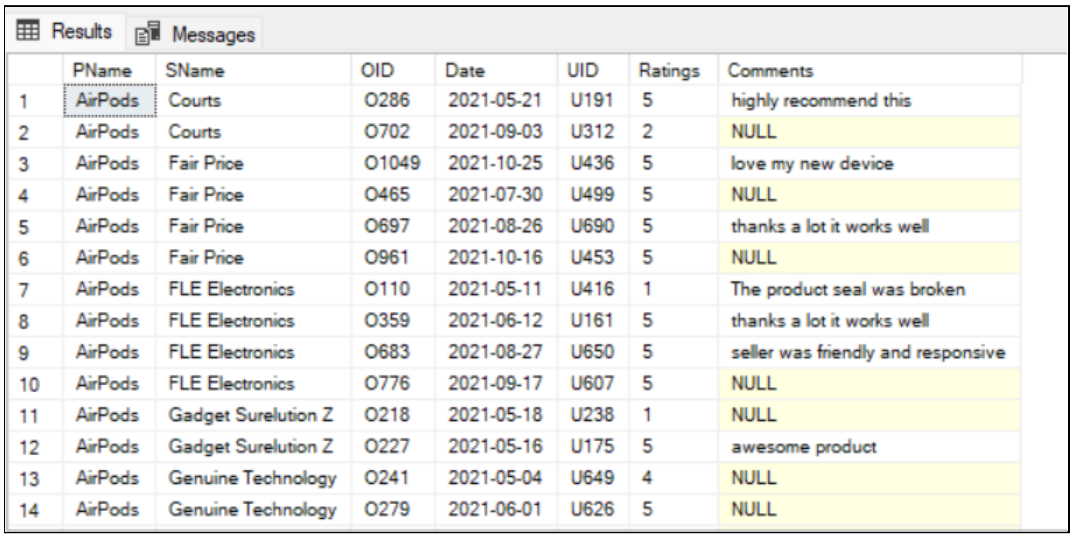


	PName	SName	Price	Start_Date	End_Date
1	AirPods	Courts	580	2021-05-01 01:43:00	2021-05-31 01:43:00
2	AirPods	Courts	602	2021-05-31 01:43:00	2021-06-30 01:43:00
3	AirPods	Courts	646	2021-06-30 01:43:00	2021-07-30 01:43:00
4	AirPods	Courts	453	2021-07-30 01:43:00	2021-08-29 01:43:00
5	AirPods	Courts	656	2021-08-29 01:43:00	2021-09-28 01:43:00
6	AirPods	Courts	558	2021-09-28 01:43:00	2021-10-31 11:08:00
7	AirPods	Fair Price	483	2021-05-01 01:43:00	2021-05-31 01:43:00
8	AirPods	Fair Price	490	2021-05-31 01:43:00	2021-06-30 01:43:00
9	AirPods	Fair Price	658	2021-06-30 01:43:00	2021-07-30 01:43:00
10	AirPods	Fair Price	548	2021-07-30 01:43:00	2021-08-29 01:43:00
11	AirPods	Fair Price	449	2021-08-29 01:43:00	2021-09-28 01:43:00
12	AirPods	Fair Price	558	2021-09-28 01:43:00	2021-10-31 11:08:00
13	AirPods	FLE Electronics	568	2021-05-01 01:43:00	2021-05-31 01:43:00
14	AirPods	FLE Electronics	664	2021-05-31 01:43:00	2021-06-30 01:43:00

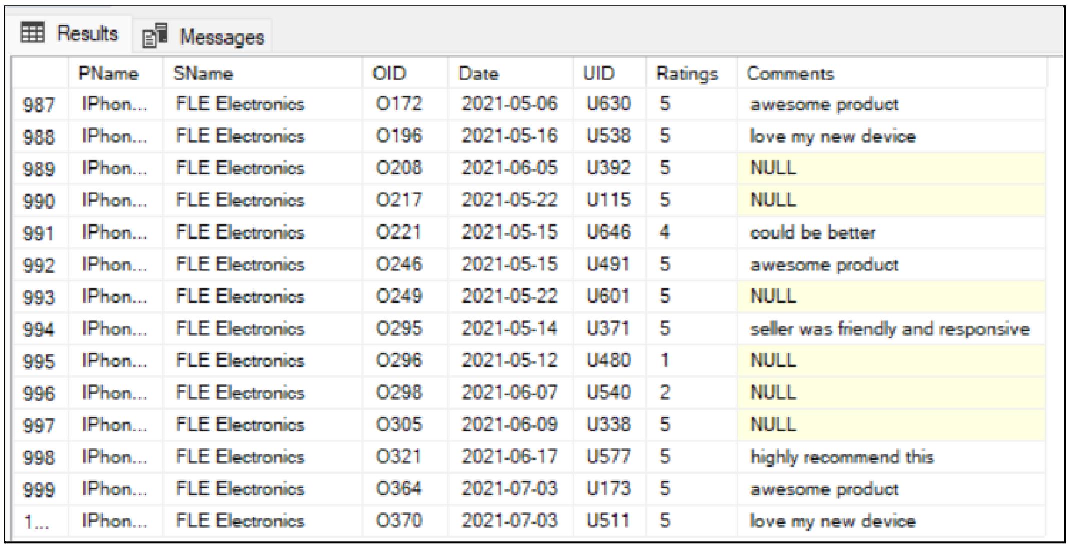


	PName	SName	Price	Start_Date	End_Date
987	Galax...	Mega Discou...	514	2021-07-30 01:43:00	2021-08-29 01:43:00
988	Galax...	Mega Discou...	561	2021-08-29 01:43:00	2021-09-28 01:43:00
989	Galax...	Mega Discou...	558	2021-09-28 01:43:00	2021-10-31 11:08:00
990	Galax...	Space Electr...	536	2021-05-01 01:43:00	2021-05-31 01:43:00
991	Galax...	Space Electr...	585	2021-05-31 01:43:00	2021-06-30 01:43:00
992	Galax...	Space Electr...	575	2021-06-30 01:43:00	2021-07-30 01:43:00
993	Galax...	Space Electr...	654	2021-07-30 01:43:00	2021-08-29 01:43:00
994	Galax...	Space Electr...	534	2021-08-29 01:43:00	2021-09-28 01:43:00
995	Galax...	Space Electr...	558	2021-09-28 01:43:00	2021-10-31 11:08:00
996	Galax...	The Arcade	636	2021-05-01 01:43:00	2021-05-31 01:43:00
997	Galax...	The Arcade	631	2021-05-31 01:43:00	2021-06-30 01:43:00
998	Galax...	The Arcade	619	2021-06-30 01:43:00	2021-07-30 01:43:00
999	Galax...	The Arcade	519	2021-07-30 01:43:00	2021-08-29 01:43:00
1...	Galax...	The Arcade	539	2021-08-29 01:43:00	2021-09-28 01:43:00

Feedback

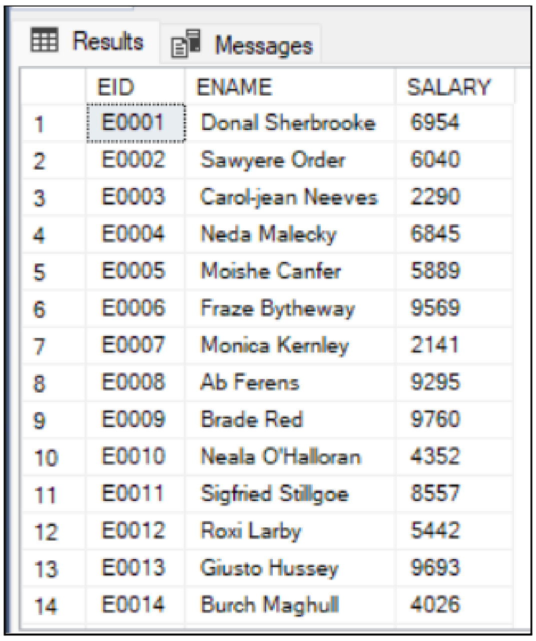


	PName	SName	OID	Date	UID	Ratings	Comments
1	AirPods	Courts	O286	2021-05-21	U191	5	highly recommend this
2	AirPods	Courts	O702	2021-09-03	U312	2	NULL
3	AirPods	Fair Price	O1049	2021-10-25	U436	5	love my new device
4	AirPods	Fair Price	O465	2021-07-30	U499	5	NULL
5	AirPods	Fair Price	O697	2021-08-26	U690	5	thanks a lot it works well
6	AirPods	Fair Price	O961	2021-10-16	U453	5	NULL
7	AirPods	FLE Electronics	O110	2021-05-11	U416	1	The product seal was broken
8	AirPods	FLE Electronics	O359	2021-06-12	U161	5	thanks a lot it works well
9	AirPods	FLE Electronics	O683	2021-08-27	U650	5	seller was friendly and responsive
10	AirPods	FLE Electronics	O776	2021-09-17	U607	5	NULL
11	AirPods	Gadget Surelution Z	O218	2021-05-18	U238	1	NULL
12	AirPods	Gadget Surelution Z	O227	2021-05-16	U175	5	awesome product
13	AirPods	Genuine Technology	O241	2021-05-04	U649	4	NULL
14	AirPods	Genuine Technology	O279	2021-06-01	U626	5	NULL



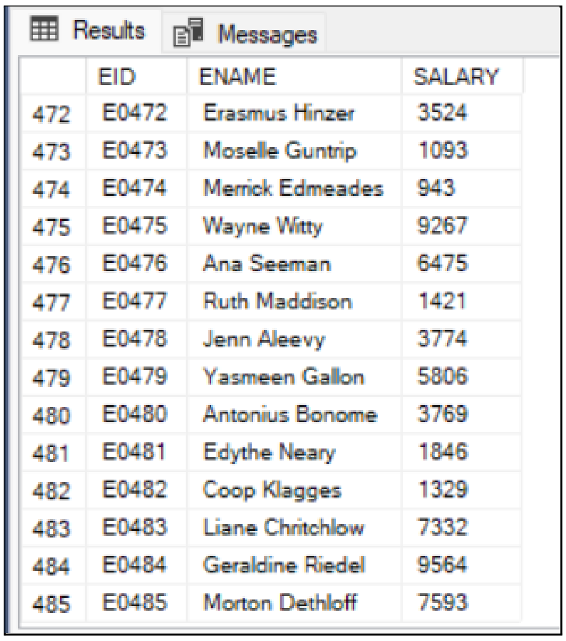
	PName	SName	OID	Date	UID	Ratings	Comments
987	IPhon...	FLE Electronics	O172	2021-05-06	U630	5	awesome product
988	IPhon...	FLE Electronics	O196	2021-05-16	U538	5	love my new device
989	IPhon...	FLE Electronics	O208	2021-06-05	U392	5	NULL
990	IPhon...	FLE Electronics	O217	2021-05-22	U115	5	NULL
991	IPhon...	FLE Electronics	O221	2021-05-15	U646	4	could be better
992	IPhon...	FLE Electronics	O246	2021-05-15	U491	5	awesome product
993	IPhon...	FLE Electronics	O249	2021-05-22	U601	5	NULL
994	IPhon...	FLE Electronics	O295	2021-05-14	U371	5	seller was friendly and responsive
995	IPhon...	FLE Electronics	O296	2021-05-12	U480	1	NULL
996	IPhon...	FLE Electronics	O298	2021-06-07	U540	2	NULL
997	IPhon...	FLE Electronics	O305	2021-06-09	U338	5	NULL
998	IPhon...	FLE Electronics	O321	2021-06-17	U577	5	highly recommend this
999	IPhon...	FLE Electronics	O364	2021-07-03	U173	5	awesome product
1...	IPhon...	FLE Electronics	O370	2021-07-03	U511	5	love my new device

Employee



Results Messages

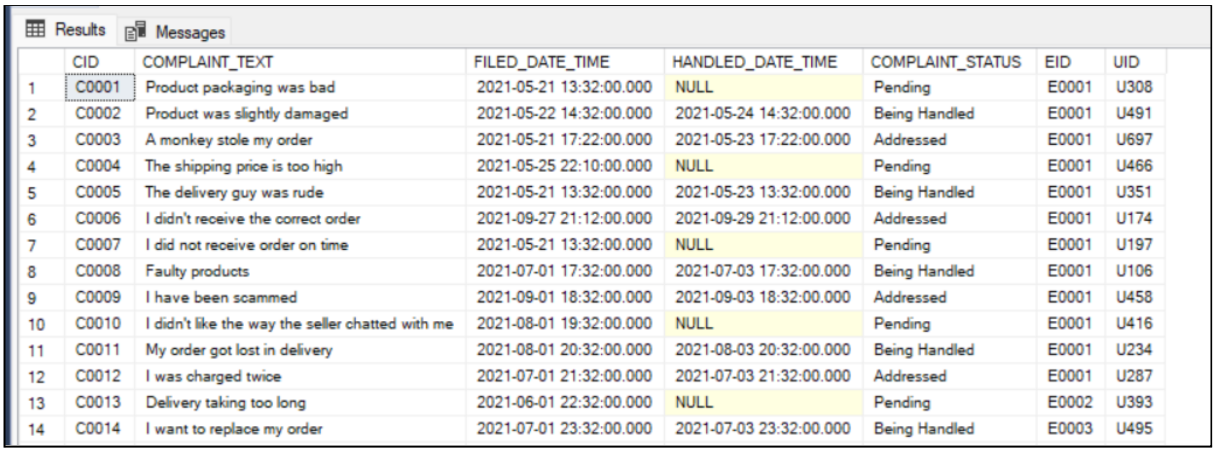
	EID	ENAME	SALARY
1	E0001	Donal Sherbrooke	6954
2	E0002	Sawyere Order	6040
3	E0003	Carol-jean Neeves	2290
4	E0004	Neda Malecky	6845
5	E0005	Moishe Canfer	5889
6	E0006	Fraze Bytheway	9569
7	E0007	Monica Kernley	2141
8	E0008	Ab Ferens	9295
9	E0009	Brade Red	9760
10	E0010	Neala O'Halloran	4352
11	E0011	Sigfried Stillgoe	8557
12	E0012	Roxi Larby	5442
13	E0013	Giusto Hussey	9693
14	E0014	Burch Maghull	4026



Results Messages

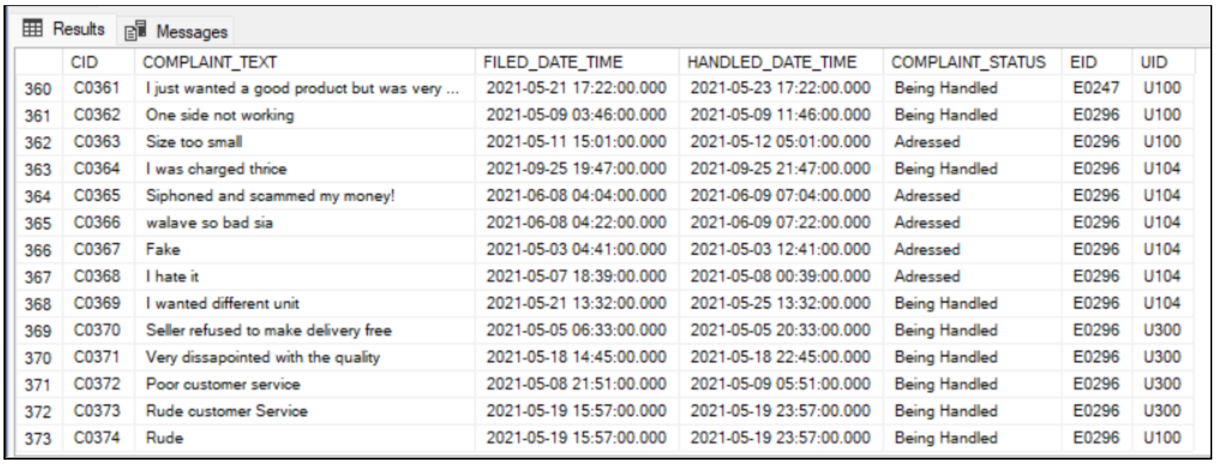
	EID	ENAME	SALARY
472	E0472	Erasmus Hinzer	3524
473	E0473	Moselle Guntrip	1093
474	E0474	Merrick Edmeades	943
475	E0475	Wayne Witty	9267
476	E0476	Ana Seeman	6475
477	E0477	Ruth Maddison	1421
478	E0478	Jenn Aleevy	3774
479	E0479	Yasmeen Gallon	5806
480	E0480	Antonius Bonome	3769
481	E0481	Edythe Neary	1846
482	E0482	Coop Klagges	1329
483	E0483	Liane Chritchlow	7332
484	E0484	Geraldine Riedel	9564
485	E0485	Morton Dethloff	7593

Complaints



Results Messages

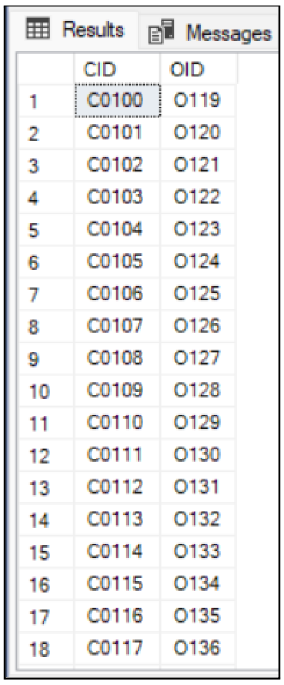
	CID	COMPLAINT_TEXT	FILED_DATE_TIME	HANDLED_DATE_TIME	COMPLAINT_STATUS	EID	UID
1	C0001	Product packaging was bad	2021-05-21 13:32:00.000	NULL	Pending	E0001	U308
2	C0002	Product was slightly damaged	2021-05-22 14:32:00.000	2021-05-24 14:32:00.000	Being Handled	E0001	U491
3	C0003	A monkey stole my order	2021-05-21 17:22:00.000	2021-05-23 17:22:00.000	Addressed	E0001	U697
4	C0004	The shipping price is too high	2021-05-25 22:10:00.000	NULL	Pending	E0001	U466
5	C0005	The delivery guy was rude	2021-05-21 13:32:00.000	2021-05-23 13:32:00.000	Being Handled	E0001	U351
6	C0006	I didn't receive the correct order	2021-09-27 21:12:00.000	2021-09-29 21:12:00.000	Addressed	E0001	U174
7	C0007	I did not receive order on time	2021-05-21 13:32:00.000	NULL	Pending	E0001	U197
8	C0008	Faulty products	2021-07-01 17:32:00.000	2021-07-03 17:32:00.000	Being Handled	E0001	U106
9	C0009	I have been scammed	2021-09-01 18:32:00.000	2021-09-03 18:32:00.000	Addressed	E0001	U458
10	C0010	I didn't like the way the seller chatted with me	2021-08-01 19:32:00.000	NULL	Pending	E0001	U416
11	C0011	My order got lost in delivery	2021-08-01 20:32:00.000	2021-08-03 20:32:00.000	Being Handled	E0001	U234
12	C0012	I was charged twice	2021-07-01 21:32:00.000	2021-07-03 21:32:00.000	Addressed	E0001	U287
13	C0013	Delivery taking too long	2021-06-01 22:32:00.000	NULL	Pending	E0002	U393
14	C0014	I want to replace my order	2021-07-01 23:32:00.000	2021-07-03 23:32:00.000	Being Handled	E0003	U495



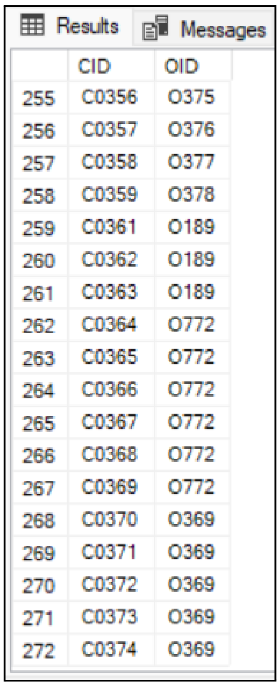
Results Messages

	CID	COMPLAINT_TEXT	FILED_DATE_TIME	HANDLED_DATE_TIME	COMPLAINT_STATUS	EID	UID
360	C0361	I just wanted a good product but was very ...	2021-05-21 17:22:00.000	2021-05-23 17:22:00.000	Being Handled	E0247	U100
361	C0362	One side not working	2021-05-09 03:46:00.000	2021-05-09 11:46:00.000	Being Handled	E0296	U100
362	C0363	Size too small	2021-05-11 15:01:00.000	2021-05-12 05:01:00.000	Adressed	E0296	U100
363	C0364	I was charged thrice	2021-09-25 19:47:00.000	2021-09-25 21:47:00.000	Being Handled	E0296	U104
364	C0365	Siphoned and scammed my money!	2021-06-08 04:04:00.000	2021-06-09 07:04:00.000	Adressed	E0296	U104
365	C0366	walave so bad sia	2021-06-08 04:22:00.000	2021-06-09 07:22:00.000	Adressed	E0296	U104
366	C0367	Fake	2021-05-03 04:41:00.000	2021-05-03 12:41:00.000	Adressed	E0296	U104
367	C0368	I hate it	2021-05-07 18:39:00.000	2021-05-08 00:39:00.000	Adressed	E0296	U104
368	C0369	I wanted different unit	2021-05-21 13:32:00.000	2021-05-25 13:32:00.000	Being Handled	E0296	U104
369	C0370	Seller refused to make delivery free	2021-05-05 06:33:00.000	2021-05-05 20:33:00.000	Being Handled	E0296	U300
370	C0371	Very dissapointed with the quality	2021-05-18 14:45:00.000	2021-05-18 22:45:00.000	Being Handled	E0296	U300
371	C0372	Poor customer service	2021-05-08 21:51:00.000	2021-05-09 05:51:00.000	Being Handled	E0296	U300
372	C0373	Rude customer Service	2021-05-19 15:57:00.000	2021-05-19 23:57:00.000	Being Handled	E0296	U300
373	C0374	Rude	2021-05-19 15:57:00.000	2021-05-19 23:57:00.000	Being Handled	E0296	U100

Complaints On Order

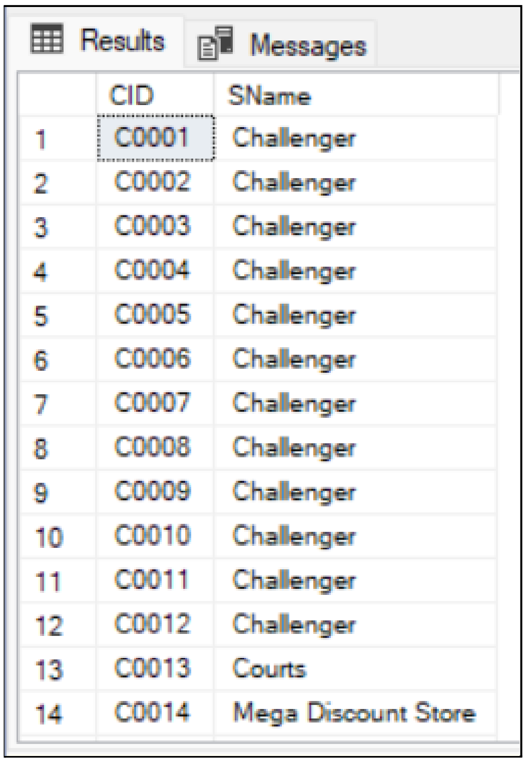


	CID	OID
1	C0100	O119
2	C0101	O120
3	C0102	O121
4	C0103	O122
5	C0104	O123
6	C0105	O124
7	C0106	O125
8	C0107	O126
9	C0108	O127
10	C0109	O128
11	C0110	O129
12	C0111	O130
13	C0112	O131
14	C0113	O132
15	C0114	O133
16	C0115	O134
17	C0116	O135
18	C0117	O136

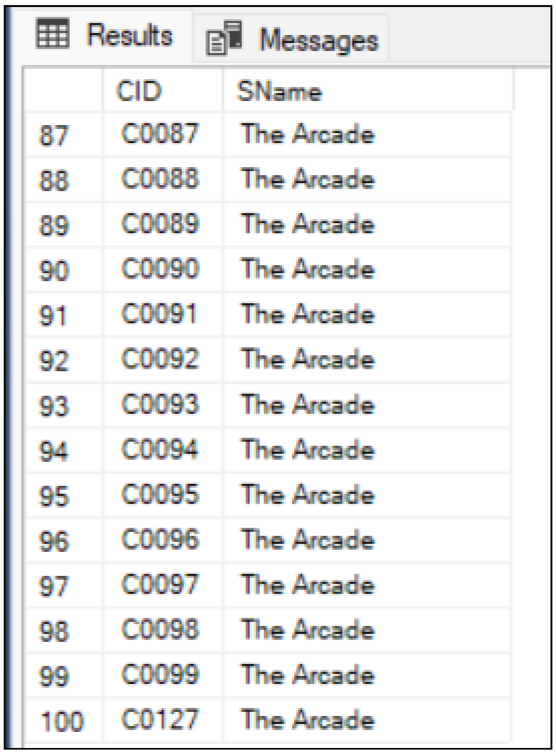


	CID	OID
255	C0356	O375
256	C0357	O376
257	C0358	O377
258	C0359	O378
259	C0361	O189
260	C0362	O189
261	C0363	O189
262	C0364	O772
263	C0365	O772
264	C0366	O772
265	C0367	O772
266	C0368	O772
267	C0369	O772
268	C0370	O369
269	C0371	O369
270	C0372	O369
271	C0373	O369
272	C0374	O369

Complaints On Shop



	CID	SName
1	C0001	Challenger
2	C0002	Challenger
3	C0003	Challenger
4	C0004	Challenger
5	C0005	Challenger
6	C0006	Challenger
7	C0007	Challenger
8	C0008	Challenger
9	C0009	Challenger
10	C0010	Challenger
11	C0011	Challenger
12	C0012	Challenger
13	C0013	Courts
14	C0014	Mega Discount Store



	CID	SName
87	C0087	The Arcade
88	C0088	The Arcade
89	C0089	The Arcade
90	C0090	The Arcade
91	C0091	The Arcade
92	C0092	The Arcade
93	C0093	The Arcade
94	C0094	The Arcade
95	C0095	The Arcade
96	C0096	The Arcade
97	C0097	The Arcade
98	C0098	The Arcade
99	C0099	The Arcade
100	C0127	The Arcade

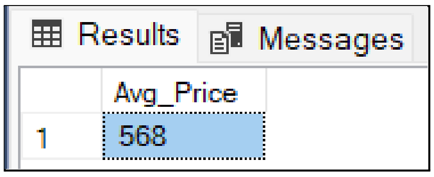
SQL Queries for Appendix B

Query 1: Find the average price of "iPhone X"s on Shiokee from 1 August 2021 to 31 August 2021.

```
WITH BothInAug AS

     (SELECT  SUM(Price*DATEDIFF(DAY,  Start_Date,  End_Date))  AS  Sigma_xn,
SUM(DATEDIFF(DAY, Start_Date, End_Date)) AS Sigma_n

      FROM PRICE_HISTORY

      WHERE PName = 'IPhone X'

            AND ((MONTH(Start_Date)='8' AND YEAR(Start_Date) = '2021') AND

            (MONTH(End_Date)='8' AND YEAR(End_Date)='2021'))),

OnlyStartInAug AS

    (SELECT SUM(Price*DATEDIFF(DAY, Start_Date, '2021/08/31')) AS Sigma_xn,
SUM(DATEDIFF(DAY, Start_Date, '2021/08/31')) AS Sigma_n

      FROM PRICE_HISTORY

      WHERE PName = 'IPhone X'

            AND ((MONTH(Start_Date)='8' AND YEAR(Start_Date) = '2021') AND

            (MONTH(End_Date) = '9' AND YEAR(End_Date) = '2021'))),

OnlyEndInAug AS

     (SELECT  SUM(Price*DATEDIFF(DAY,  '2021/08/01',  End_Date))  AS  Sigma_xn,
SUM(DATEDIFF(DAY, '2021/08/01', End_Date)) AS Sigma_n

      FROM PRICE_HISTORY

      WHERE PName = 'IPhone X'

            AND ((MONTH(Start_Date)='7' AND YEAR(Start_Date) = '2021') AND

            (MONTH(End_Date) = '8' AND YEAR(End_Date) = '2021')))

SELECT SUM(Sigma_xn) / SUM(Sigma_n) AS Avg_Price

FROM   (SELECT * FROM BothInAug

            UNION

            SELECT * FROM OnlyStartInAug

            UNION

            SELECT * FROM OnlyEndInAug) AS WeightedSums;
```

Explanation:
We have implemented a weighted average for the prices of IPhone X weighted on the number of days in August where the corresponding price was used. The formula for weighted Average is: $\mu = \Sigma\,(x.n)/\,\Sigma\,n$. Here x is the price and n is the number of days in august 2021. In order to find the number of days in August 2021 we have 3 cases:

Case 1: Both Start_Date and End_Date are in August (BothInAug)
Here, n = End_Date - Start_Date.
Case 2: Only Start_Date is in August (OnlyStartInAug)
Here, n = '2021/08/31' - Start_Date.
Case 3: Only End_Date is in August (OnlyEndInAug)
Here, n = End_Date - '2021/08/01'.
We found $\Sigma\,(x.n)$ and $\Sigma\,n$ for all 3 cases. We then merged them together (using UNION) and found the average.

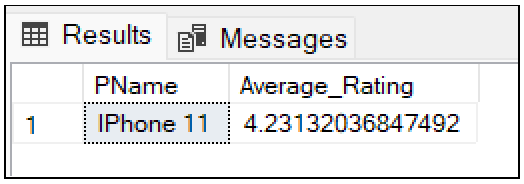


	Avg_Price
1	568

Query 2: Find products that received at least 100 ratings of "5" in August 2021, and order them by their average ratings.

```
WITH Cnt5Star AS
(SELECT PName, COUNT(Ratings) AS cnt
      FROM [ss4g3].[dbo].[Feedback]
      WHERE Ratings = 5 AND MONTH(Date) = 8
      GROUP BY PName)
SELECT f.PName, AVG(CAST(f.Ratings AS float)) AS Average_Rating
FROM [ss4g3].[dbo].[Feedback] AS f, Cnt5Star AS cnt5
WHERE f.PName = cnt5.PName AND cnt5.cnt >= 100
GROUP BY f.PName
ORDER BY AVG(f.Ratings), f.PName;
```

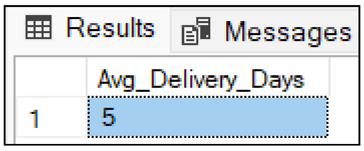
Explanation:
We first create a view wherein we select all the PNames and Ratings which are 5 for the month of August and then group them by PName from the Feedback table. Then, we select the Average_Rating of those products from Feedback table which have been rated 5 stars more than or equal to 100 times



	PName	Average_Rating
1	IPhone 11	4.23132036847492

Query 3: For all products purchased in June 2021 that have been delivered, find the average time from the ordering date to the delivery date.

```
/* Assumption: 1) Average Delivery Time measured in Days
               2) We want average delivery time as a whole. */
SELECT AVG(DATEDIFF(DAY, ORDERS.Date_Time, DeliveryDate)) AS
Avg_Delivery_Days
FROM PRODUCTS_IN_ORDER, ORDERS
WHERE PRODUCTS_IN_ORDER.OID = ORDERS.OID AND
     OrderStatus = 'Delivered' AND
      MONTH(ORDERS.Date_Time)='6' AND MONTH(DeliveryDate)='6' AND
      YEAR(ORDERS.Date_Time)='2021' AND YEAR(DeliveryDate)='2021';
```

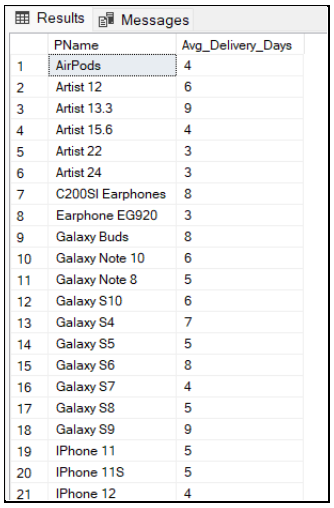


	Avg_Delivery_Days
1	5

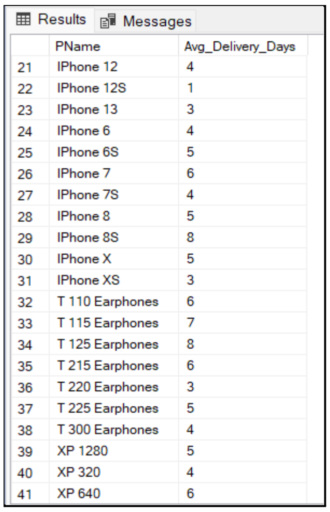
Explanation:
We joined tha Products_in_Order and Orders table using OID as the common attribute. We then selected the records from August 2021 having OrderStatus as “Delivered” . Then we calculated the average delivery time using the difference between Delivery Date (Products_in_Order.DeliveryDate) and Order Date(Orders.Date_Time)

```
/* Assumption:1) Average Delivery Time measured in Days
              2) We want an average delivery time for each product*/

SELECT PName, AVG(DATEDIFF(DAY, ORDERS.Date_Time, DeliveryDate)) AS
Avg_Delivery_Days
FROM PRODUCTS_IN_ORDER, ORDERS
WHERE PRODUCTS_IN_ORDER.OID = ORDERS.OID AND
OrderStatus = 'Delivered' AND
      MONTH(ORDERS.Date_Time) = '6' AND MONTH(DeliveryDate) ='6' AND
      YEAR(ORDERS.Date_Time) = '2021' AND YEAR(DeliveryDate) ='2021'
GROUP BY PName;
```

Explanation: Same as the first part but for each product.



	PName	Avg_Delivery_Days
1	AirPods	4
2	Artist 12	6
3	Artist 13.3	9
4	Artist 15.6	4
5	Artist 22	3
6	Artist 24	3
7	C200SI Earphones	8
8	Earphone EG920	3
9	Galaxy Buds	8
10	Galaxy Note 10	6
11	Galaxy Note 8	5
12	Galaxy S10	6
13	Galaxy S4	7
14	Galaxy S5	5
15	Galaxy S6	8
16	Galaxy S7	4
17	Galaxy S8	5
18	Galaxy S9	9
19	IPhone 11	5
20	IPhone 11S	5
21	IPhone 12	4

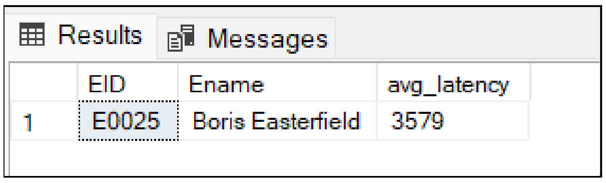


	PName	Avg_Delivery_Days
21	IPhone 12	4
22	IPhone 12S	1
23	IPhone 13	3
24	IPhone 6	4
25	IPhone 6S	5
26	IPhone 7	6
27	IPhone 7S	4
28	IPhone 8	5
29	IPhone 8S	8
30	IPhone X	5
31	IPhone XS	3
32	T 110 Earphones	6
33	T 115 Earphones	7
34	T 125 Earphones	8
35	T 215 Earphones	6
36	T 220 Earphones	3
37	T 225 Earphones	5
38	T 300 Earphones	4
39	XP 1280	5
40	XP 320	4
41	XP 640	6

Query 4: Let us define the "latency" of an employee by the average that he/she takes to process a complaint. Find the employee with the smallest latency.

```
/* Assumption: No two employees can have the same exact latency
accurate to seconds */
SELECT TOP 1 Employee.EID,Employee.Ename, X.avg_latency
FROM
    (SELECT Employee.EID, AVG(DATEDIFF(second, Filed_Date_Time,
Handled_Date_Time)) as avg_latency
    FROM Complaints,Employee
    WHERE Employee.EID = Complaints.EID
    GROUP BY Employee.EID) AS X, Employee
WHERE Employee.EID= X.EID AND X.avg_latency IS NOT NULL
ORDER BY avg_latency ASC
```

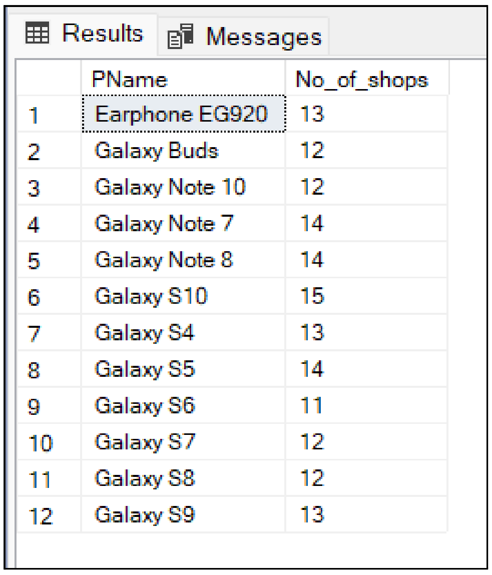
Explanation:
First, we join the Complaints and the Employee tables using EID as the common attribute. Then, we calculate the average latency for each employee by finding the average of the difference between Filed_Date_Time and Handled_Date_Time. Then we sort the table in ascending order, using the avg_latency and select the 1st record from this table. We have assumed that no 2 employees can have the same latency. We have calculated the latency in seconds. Hence there is only 1 employee with the smallest latency.



	EID	Ename	avg_latency
1	E0025	Boris Easterfield	3579

Query 5: Produce a list that contains (i) all products made by Samsung, and (ii) for each of them, the number of shops on Shiokee that sell the product.

```
SELECT a.PName,COUNT(DISTINCT SName) AS No_of_shops
FROM PRODUCTS_IN_SHOPS AS a,PRODUCTS
WHERE a.PName IN (SELECT PName FROM PRODUCTS
                  WHERE Maker='Samsung')
GROUP BY a.PName;
```



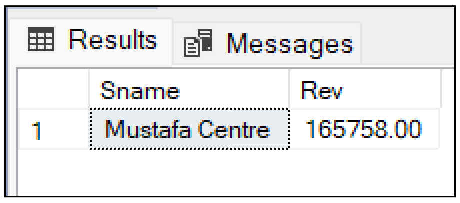
	PName	No_of_shops
1	Earphone EG920	13
2	Galaxy Buds	12
3	Galaxy Note 10	12
4	Galaxy Note 7	14
5	Galaxy Note 8	14
6	Galaxy S10	15
7	Galaxy S4	13
8	Galaxy S5	14
9	Galaxy S6	11
10	Galaxy S7	12
11	Galaxy S8	12
12	Galaxy S9	13

Explanation: First we select all products from PRODUCTS which are made by Samsung. Then we group by product name and count the shops with distinct shop names for each product

Query 6: Find shops that made the most revenue in August 2021.

```sql
/* Assumption: Payment is released to the Shop by Shiokee only
after product is delivered (same as lazada) */
WITH RevShop AS
       (SELECT Sname, SUM(OPrice * OQuantity) AS Rev
        FROM [ss4g3].[dbo].[PRODUCTS_IN_ORDER]
        WHERE MONTH(DeliveryDate) = '8'
        GROUP BY SName)
SELECT Sname, Rev
FROM RevShop
WHERE Rev = (SELECT MAX(Rev)
                 FROM RevShop);
```

Explanation:
We start by finding the revenue made by each shop in August. This is stored in RevShop. We use DeliveryDate as payment is released to the shop after the product is delivered. Next we select all the shops in RevShop where the Revenue(Rev) is equal to the max revenue in RevShop. This makes sure that if multiple shops are tied for highest revenue they all show up.

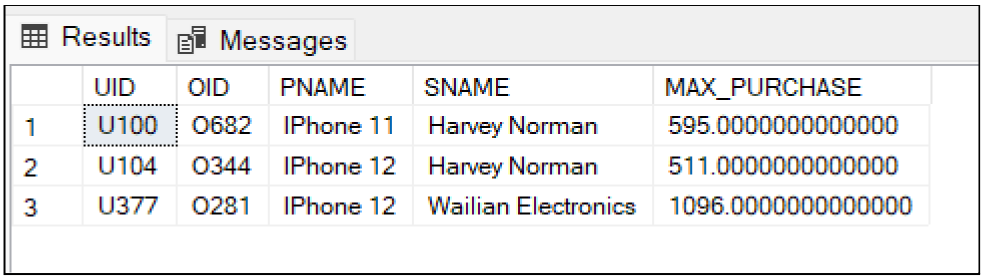


	Sname	Rev
1	Mustafa Centre	165758.00

Query 7: For users that made the most amount of complaints, find the most expensive products he/she has ever purchased.

```
DROP VIEW USERID,USERORDERS,PRODPRICES
GO
CREATE VIEW USERID AS

SELECT UID,MAX (P.OPrice/P.OQuantity) As MAX_PURCHASE
        FROM
            /* OID of users who made most amount of complaints */
            (SELECT O.OID, X.UID
             FROM
                (SELECT UID, COUNT (*) AS MAX_COMPLAINT
                 FROM COMPLAINTS AS C
                 GROUP BY UID
                 HAVING COUNT(*)=
                /* Displays UID with max number of complaints*/
                (SELECT MAX(Y.COMPLAINT_COUNT)
                 FROM
                    /*Counts number of complaints for each UID */
                    (SELECT COUNT(CID) AS COMPLAINT_COUNT
                     FROM COMPLAINTS
                     GROUP BY UID) AS Y ))AS X, ORDERS AS O
            WHERE O.UID=X.UID) AS Z, PRODUCTS_IN_ORDER AS P
        WHERE Z.OID=P.OID
        GROUP BY UID
GO

CREATE VIEW USERORDERS AS
SELECT DISTINCT USERID.UID, OID
FROM USERID, ORDERS AS O
WHERE USERID.UID = O.UID
GO

CREATE VIEW PRODPRICES AS
SELECT O.UID,O.OID,PNAME, SNAME,OPRICE/OQUANTITY AS PRICE
FROM USERORDERS AS O,PRODUCTS_IN_ORDER AS P
WHERE O.OID = P.OID
GO

SELECT U.UID,OID,PNAME,SNAME,MAX_PURCHASE
FROM PRODPRICES AS P,USERID AS U
WHERE P.UID=U.UID AND P.PRICE = U.MAX_PURCHASE
```

Explanation: First we count the number of complaints for each user. Then, we select the UID of those users with the maximum number of complaints. Using the Orders table,we find the OID of these users. Using the Products_In_Orders table, we find the products purchased by these users. Then, we select the maximum purchase for each user.

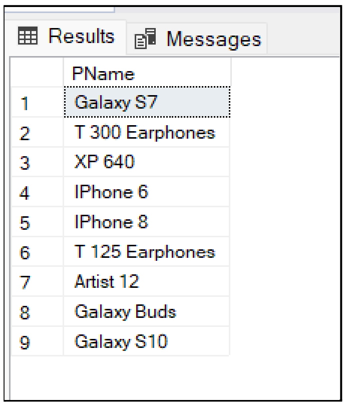


Results | Messages

	UID	OID	PNAME	SNAME	MAX_PURCHASE
1	U100	O682	IPhone 11	Harvey Norman	595.0000000000000
2	U104	O344	IPhone 12	Harvey Norman	511.0000000000000
3	U377	O281	IPhone 12	Wailian Electronics	1096.0000000000000

Query 8: Find products that have never been purchased by some users, but are the top 5 most purchased products by other users in August 2021.

```
SELECT DISTINCT pname
FROM products
WHERE pname NOT IN (
/* Product that has never been purchased by SOME users */
    SELECT pname
    FROM orders, products_in_order
    GROUP BY pname
/* DISTINCT because some users may purchase same products multiple times
*/
    HAVING
      Count(DISTINCT uid) = (
        /* get the number of users */
        SELECT Count(uid)
        FROM users
      )
  )
  AND pname IN (
    /* TOP 5 products in Aug, 2021*/
    SELECT TOP 5 WITH ties pname
    FROM products_in_order
    WHERE Month(deliverydate) = 8
    GROUP BY pname
    ORDER BY Sum(oquantity)
  );
```



Results | Messages

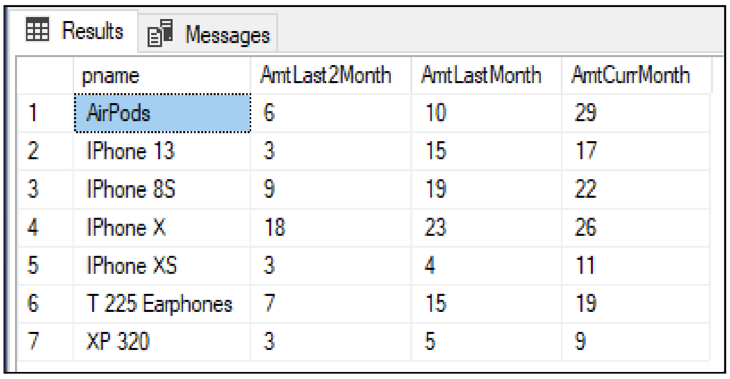
	PName
1	Galaxy S7
2	T 300 Earphones
3	XP 640
4	IPhone 6
5	IPhone 8
6	T 125 Earphones
7	Artist 12
8	Galaxy Buds
9	Galaxy S10

Explanation:
First we find the products that have been purchased by all users. Then we find products from PRODUCTS that are not returned in the previous query.Then we find the top 5 products in August 2021, and then select those products that are common between the two.

Query 9: Find products that are increasingly being purchased over at least 3 months.

```
/*Assumption:
1) Today's Date is 31 Oct 2021
2) October is included in past 3 months since it is almost over
3) Product is considered sold when order is placed */

DECLARE @date date = '2021/10/31';

WITH prodamtaug
     AS (SELECT pname,SUM(oquantity) AS AmtLast2Month
         FROM   products_in_order, orders
         WHERE  orders.oid = products_in_order.oid
                AND DATEDIFF(MONTH, orders.date_time, @date) = 2
         GROUP  BY pname),
     prodamtsept
     AS (SELECT pname,SUM(oquantity) AS AmtLastMonth
         FROM   products_in_order, orders
         WHERE  orders.oid = products_in_order.oid
                AND DATEDIFF(MONTH, orders.date_time, @date) = 1
         GROUP  BY pname),
     prodamtoct
     AS (SELECT pname,SUM(oquantity) AS AmtCurrMonth
         FROM   products_in_order,orders
         WHERE  orders.oid = products_in_order.oid
                AND DATEDIFF(MONTH, orders.date_time, @date) = 0
         GROUP  BY pname)

SELECT prodamtsept.pname, AmtLast2Month, AmtLastMonth, AmtCurrMonth
FROM   prodamtaug, prodamtsept, prodamtoct
WHERE  prodamtaug.pname = prodamtsept.pname
       AND prodamtsept.pname = prodamtoct.pname
       AND AmtLast2Month < AmtLastMonth
       AND AmtLastMonth < AmtCurrMonth;
```

Explanation: First we find the amount of each product sold in August (ProdAmtAug), September (ProdAmtSept) and October (ProdAmtOct). We then join them over PName. We then select only those products where the monthly sales have been increasing over the past months, i.e, the August Sales < September Sales < October Sales.



	pname	AmtLast2Month	AmtLastMonth	AmtCurrMonth
1	AirPods	6	10	29
2	IPhone 13	3	15	17
3	IPhone 8S	9	19	22
4	IPhone X	18	23	26
5	IPhone XS	3	4	11
6	T 225 Earphones	7	15	19
7	XP 320	3	5	9

Additional Effort and Explanations

1) TinyInt is used for ratings in FEEDBACK to optimize storage space.
2) Values of OPrice in PRODUCTS_IN_ORDER match that of PRICE_HISTORY for the given period of time.
3) Comments in FEEDBACK can be NULL in order to reflect real world data.
4) UID was added to FEEDBACK in order to make it easier to look up the user who made the query (on the recommendation of our TA).
5) ON DELETE and ON UPDATE is set to CASCADE for all foreign keys* in order to make the database more flexible and easy to modify.
6) Tables populated such that iStudio only sells products made by Apple.
7) The addresses used are real HDB addresses in Singapore.

*All except UID in FEEDBACK, where they are set to NO ACTION since that is already linked to OID and the relation was just added for ease of looking up the user.

FEEDBACK table creation python code

```
import sys
import random
import pandas as pd
from datetime import *
```

```
pino = pd.read_csv("Products_in_orders.csv")
#pino['DelDate'] = pd.to_datetime(pino['DelDate'])
## mm/dd/yyyy to date object
for m in range(len(pino)):
    pino['DelDate'][m] = datetime.strptime(pino['DelDate'][m], "%d-%m-%Y %H.%M").date()
```

```
orders = pd.read_csv("Orders.csv")
```

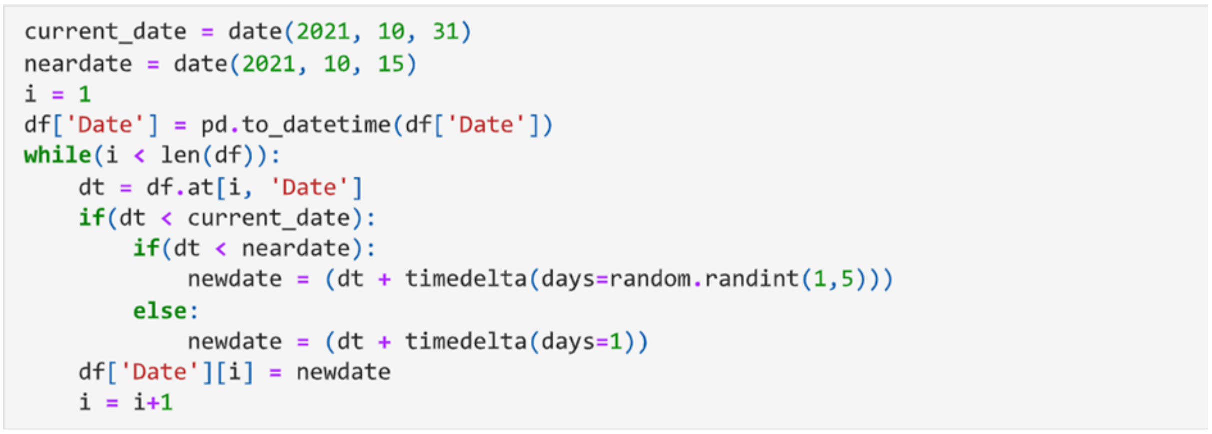
```
df = pd.DataFrame(columns = ["PName","SName","OID"])
```

```
for i in range(len(orders)):
    UID[orders.iloc[i].OID]=orders.iloc[i].UID
```

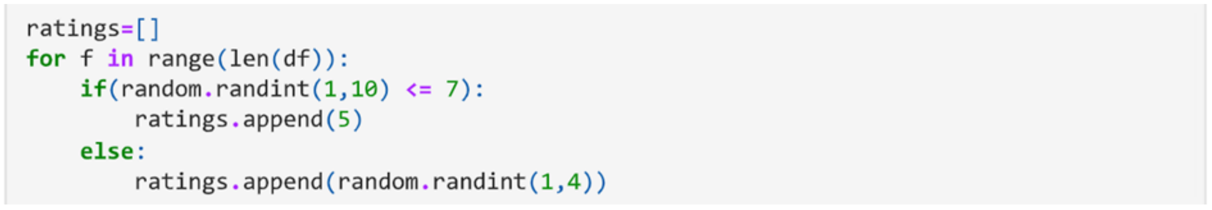
```
df = pino.copy()
df = df.rename(columns={'DelDate': 'Date',})
```

```
data=[]
for x in pino.OID:
    data.append(UID[x])
```

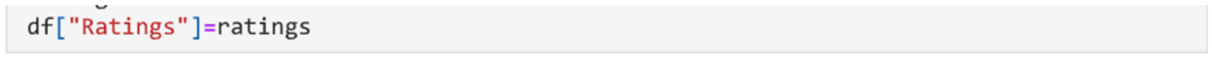
```
df['UID']=data
```



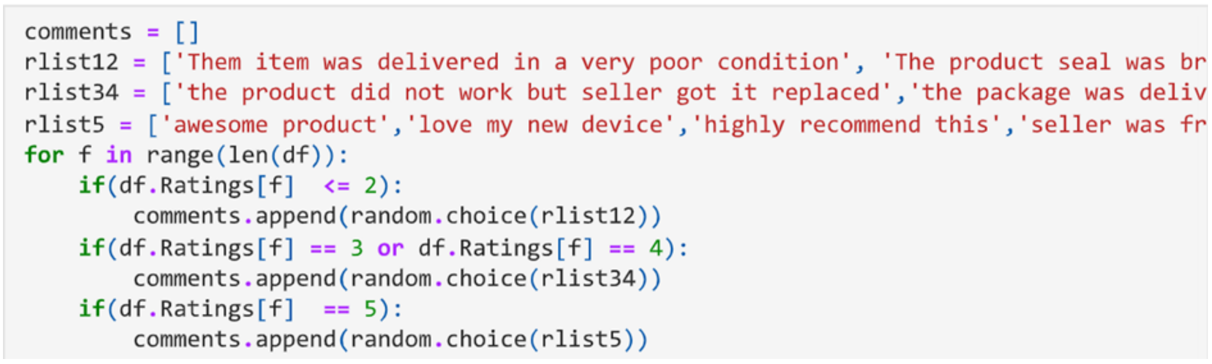
```python
current_date = date(2021, 10, 31)
neardate = date(2021, 10, 15)
i = 1
df['Date'] = pd.to_datetime(df['Date'])
while(i < len(df)):
    dt = df.at[i, 'Date']
    if(dt < current_date):
        if(dt < neardate):
            newdate = (dt + timedelta(days=random.randint(1,5)))
        else:
            newdate = (dt + timedelta(days=1))
    df['Date'][i] = newdate
    i = i+1
```



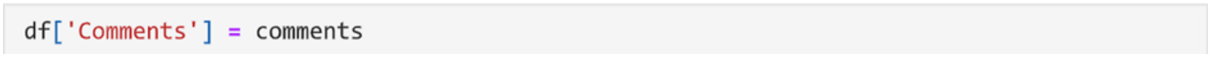
```python
ratings=[]
for f in range(len(df)):
    if(random.randint(1,10) <= 7):
        ratings.append(5)
    else:
        ratings.append(random.randint(1,4))
```



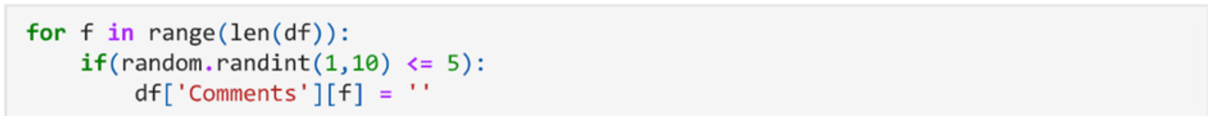
```python
df["Ratings"]=ratings
```



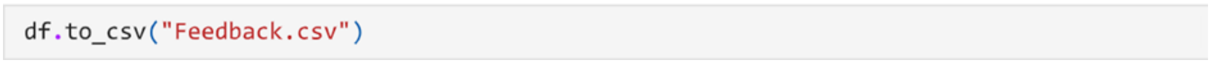
```python
comments = []
rlist12 = ['Them item was delivered in a very poor condition', 'The product seal was br
rlist34 = ['the product did not work but seller got it replaced','the package was deliv
rlist5 = ['awesome product','love my new device','highly recommend this','seller was fr
for f in range(len(df)):
    if(df.Ratings[f]  <= 2):
        comments.append(random.choice(rlist12))
    if(df.Ratings[f] == 3 or df.Ratings[f] == 4):
        comments.append(random.choice(rlist34))
    if(df.Ratings[f]  == 5):
        comments.append(random.choice(rlist5))
```



```python
df['Comments'] = comments
```



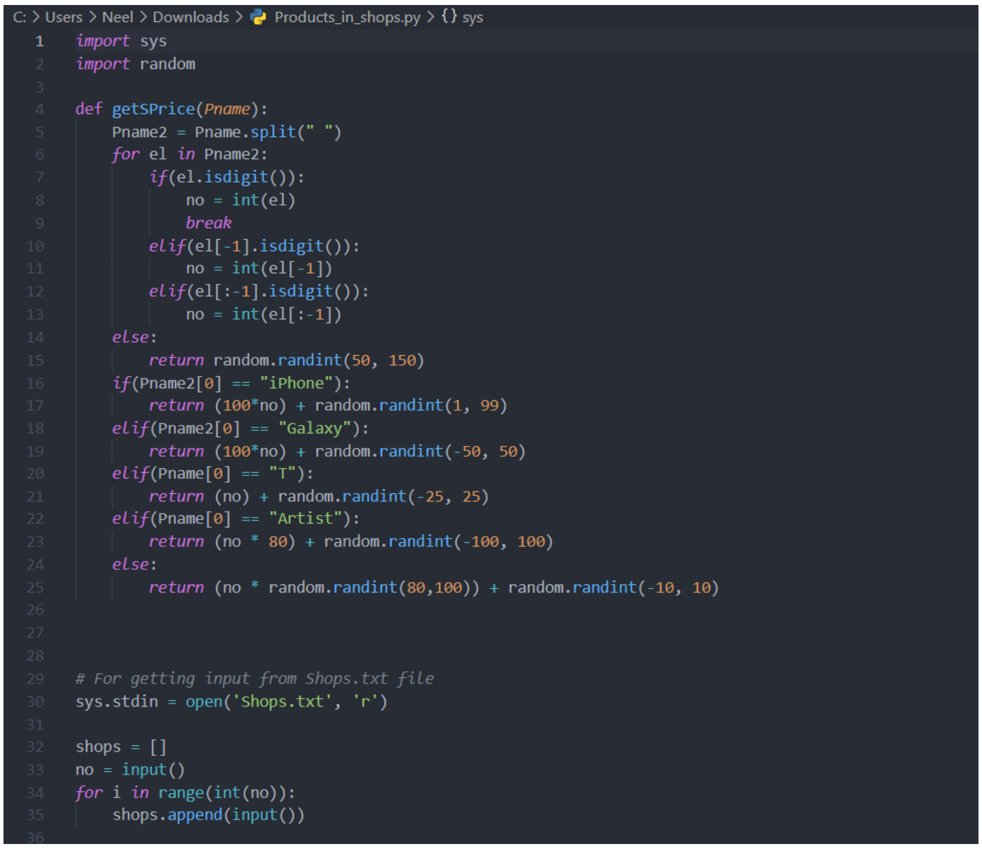
```python
for f in range(len(df)):
    if(random.randint(1,10) <= 5):
        df['Comments'][f] = ''
```



```python
df.to_csv("Feedback.csv")
```

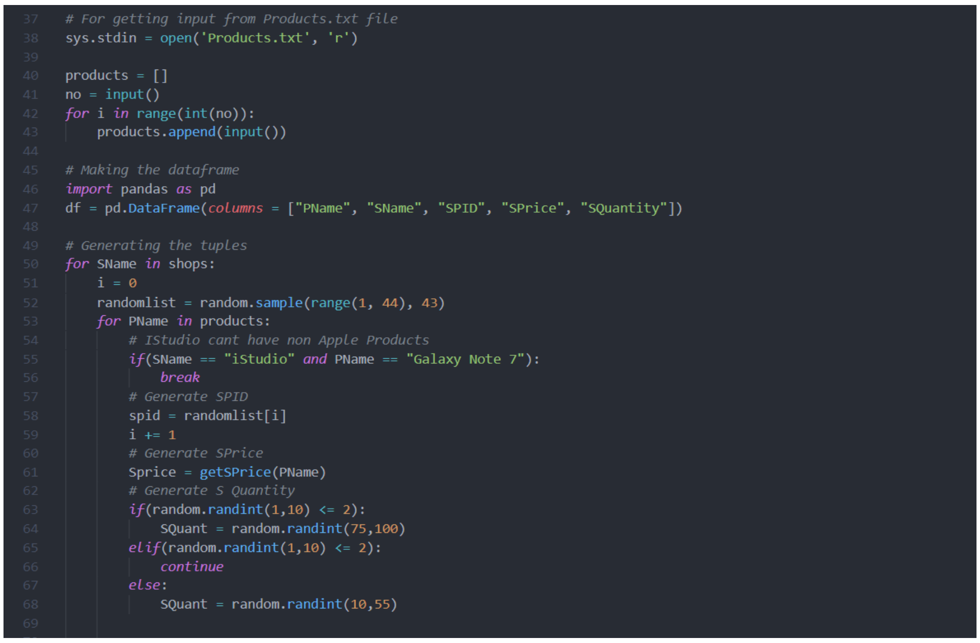
Moreover, python scripts were also made and ran for creating products in order & products in shop since we needed to maintain various dependency constraints while simultaneously having a large enough database to accommodate different possible queries. The screenshot of the code used for the aforementioned is attached below.

Products in Shop:

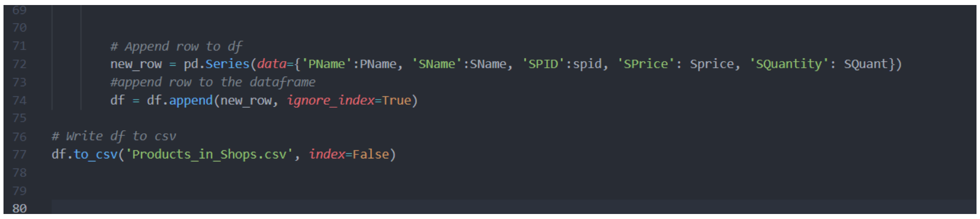


C: > Users > Neel > Downloads > Products_in_shops.py > {} sys

```python
import sys
import random

def getSPrice(Pname):
    Pname2 = Pname.split(" ")
    for el in Pname2:
        if(el.isdigit()):
            no = int(el)
            break
        elif(el[-1].isdigit()):
            no = int(el[-1])
        elif(el[:-1].isdigit()):
            no = int(el[:-1])
    else:
        return random.randint(50, 150)
    if(Pname2[0] == "iPhone"):
        return (100*no) + random.randint(1, 99)
    elif(Pname2[0] == "Galaxy"):
        return (100*no) + random.randint(-50, 50)
    elif(Pname[0] == "T"):
        return (no) + random.randint(-25, 25)
    elif(Pname[0] == "Artist"):
        return (no * 80) + random.randint(-100, 100)
    else:
        return (no * random.randint(80,100)) + random.randint(-10, 10)



# For getting input from Shops.txt file
sys.stdin = open('Shops.txt', 'r')

shops = []
no = input()
for i in range(int(no)):
    shops.append(input())

```

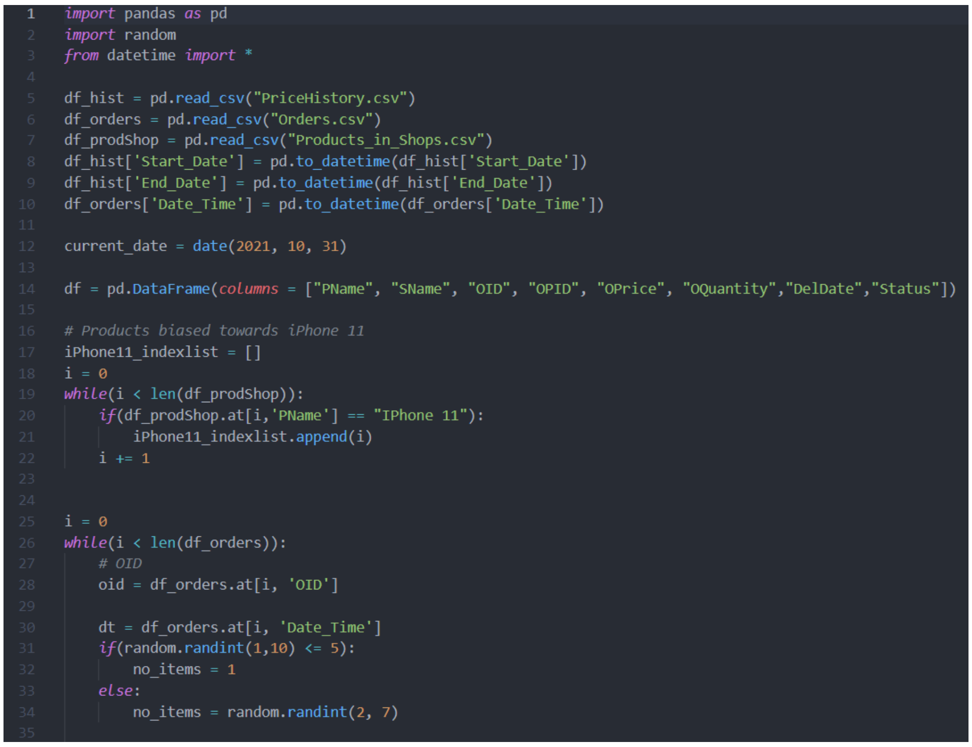


```python
# For getting input from Products.txt file
sys.stdin = open('Products.txt', 'r')

products = []
no = input()
for i in range(int(no)):
    products.append(input())

# Making the dataframe
import pandas as pd
df = pd.DataFrame(columns = ["PName", "SName", "SPID", "SPrice", "SQuantity"])

# Generating the tuples
for SName in shops:
    i = 0
    randomlist = random.sample(range(1, 44), 43)
    for PName in products:
        # IStudio cant have non Apple Products
        if(SName == "iStudio" and PName == "Galaxy Note 7"):
            break
        # Generate SPID
        spid = randomlist[i]
        i += 1
        # Generate SPrice
        Sprice = getSPrice(PName)
        # Generate S Quantity
        if(random.randint(1,10) <= 2):
            SQuant = random.randint(75,100)
        elif(random.randint(1,10) <= 2):
            continue
        else:
            SQuant = random.randint(10,55)

```

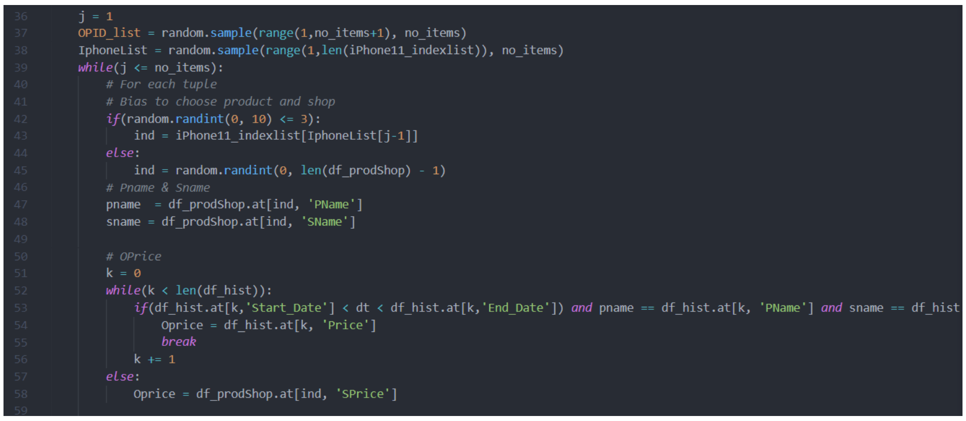


```python


        # Append row to df
        new_row = pd.Series(data={'PName':PName, 'SName':SName, 'SPID':spid, 'SPrice': Sprice, 'SQuantity': SQuant})
        #append row to the dataframe
        df = df.append(new_row, ignore_index=True)

# Write df to csv
df.to_csv('Products_in_Shops.csv', index=False)


```

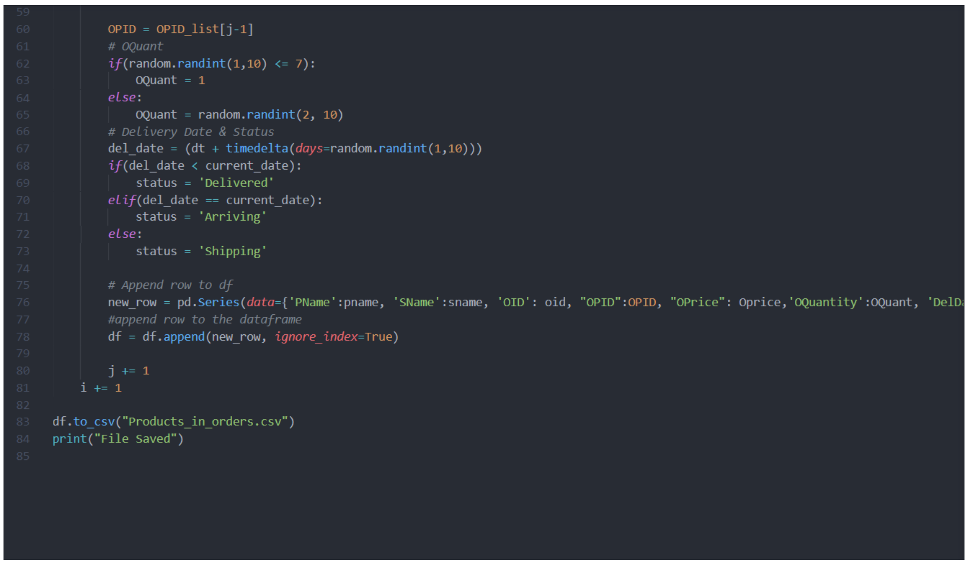
Products in Orders:



```python
import pandas as pd
import random
from datetime import *

df_hist = pd.read_csv("PriceHistory.csv")
df_orders = pd.read_csv("Orders.csv")
df_prodShop = pd.read_csv("Products_in_Shops.csv")
df_hist['Start_Date'] = pd.to_datetime(df_hist['Start_Date'])
df_hist['End_Date'] = pd.to_datetime(df_hist['End_Date'])
df_orders['Date_Time'] = pd.to_datetime(df_orders['Date_Time'])

current_date = date(2021, 10, 31)

df = pd.DataFrame(columns = ["PName", "SName", "OID", "OPID", "OPrice", "OQuantity","DelDate","Status"])

# Products biased towards iPhone 11
iPhone11_indexlist = []
i = 0
while(i < len(df_prodShop)):
    if(df_prodShop.at[i,'PName'] == "IPhone 11"):
        iPhone11_indexlist.append(i)
    i += 1


i = 0
while(i < len(df_orders)):
    # OID
    oid = df_orders.at[i, 'OID']

    dt = df_orders.at[i, 'Date_Time']
    if(random.randint(1,10) <= 5):
        no_items = 1
    else:
        no_items = random.randint(2, 7)

```

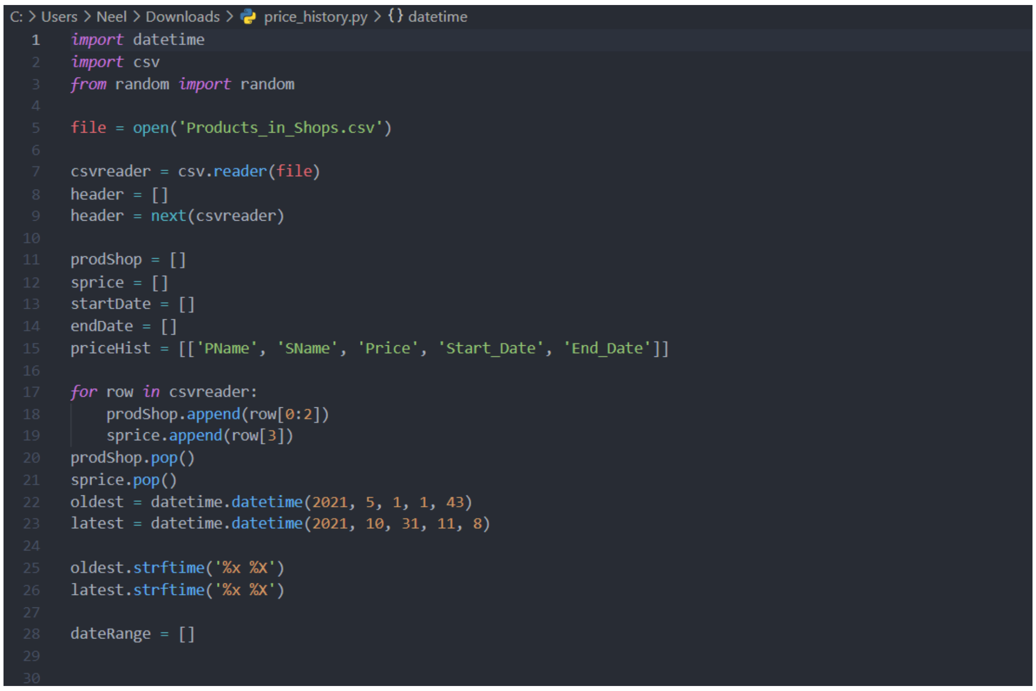


```python
    j = 1
    OPID_list = random.sample(range(1,no_items+1), no_items)
    IphoneList = random.sample(range(1,len(iPhone11_indexlist)), no_items)
    while(j <= no_items):
        # For each tuple
        # Bias to choose product and shop
        if(random.randint(0, 10) <= 3):
            ind = iPhone11_indexlist[IphoneList[j-1]]
        else:
            ind = random.randint(0, len(df_prodShop) - 1)
        # Pname & Sname
        pname  = df_prodShop.at[ind, 'PName']
        sname = df_prodShop.at[ind, 'SName']

        # OPrice
        k = 0
        while(k < len(df_hist)):
            if(df_hist.at[k,'Start_Date'] < dt < df_hist.at[k,'End_Date']) and pname == df_hist.at[k, 'PName'] and sname == df_hist
                Oprice = df_hist.at[k, 'Price']
                break
            k += 1
        else:
            Oprice = df_prodShop.at[ind, 'SPrice']

```

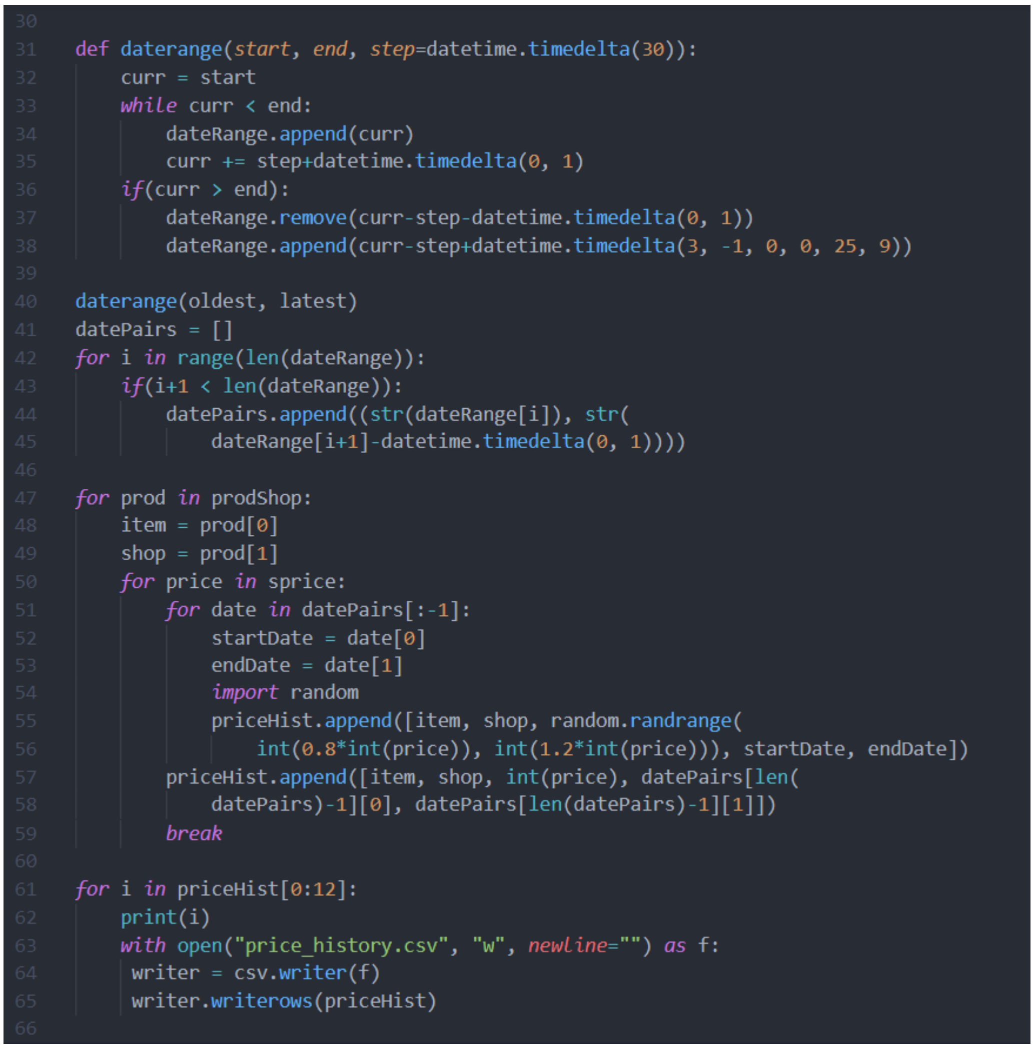


```python

        OPID = OPID_list[j-1]
        # OQuant
        if(random.randint(1,10) <= 7):
            OQuant = 1
        else:
            OQuant = random.randint(2, 10)
        # Delivery Date & Status
        del_date = (dt + timedelta(days=random.randint(1,10)))
        if(del_date < current_date):
            status = 'Delivered'
        elif(del_date == current_date):
            status = 'Arriving'
        else:
            status = 'Shipping'

        # Append row to df
        new_row = pd.Series(data={'PName':pname, 'SName':sname, 'OID': oid, "OPID":OPID, "OPrice": Oprice,'OQuantity':OQuant, 'DelD
        #append row to the dataframe
        df = df.append(new_row, ignore_index=True)

        j += 1
    i += 1

df.to_csv("Products_in_orders.csv")
print("File Saved")

```

Price History


C: > Users > Neel > Downloads > price_history.py > {} datetime

```python
import datetime
import csv
from random import random

file = open('Products_in_Shops.csv')

csvreader = csv.reader(file)
header = []
header = next(csvreader)

prodShop = []
sprice = []
startDate = []
endDate = []
priceHist = [['PName', 'SName', 'Price', 'Start_Date', 'End_Date']]

for row in csvreader:
    prodShop.append(row[0:2])
    sprice.append(row[3])
prodShop.pop()
sprice.pop()
oldest = datetime.datetime(2021, 5, 1, 1, 43)
latest = datetime.datetime(2021, 10, 31, 11, 8)

oldest.strftime('%x %X')
latest.strftime('%x %X')

dateRange = []

```



```python

def daterange(start, end, step=datetime.timedelta(30)):
    curr = start
    while curr < end:
        dateRange.append(curr)
        curr += step+datetime.timedelta(0, 1)
    if(curr > end):
        dateRange.remove(curr-step-datetime.timedelta(0, 1))
        dateRange.append(curr-step+datetime.timedelta(3, -1, 0, 0, 25, 9))

daterange(oldest, latest)
datePairs = []
for i in range(len(dateRange)):
    if(i+1 < len(dateRange)):
        datePairs.append((str(dateRange[i]), str(
            dateRange[i+1]-datetime.timedelta(0, 1))))

for prod in prodShop:
    item = prod[0]
    shop = prod[1]
    for price in sprice:
        for date in datePairs[:-1]:
            startDate = date[0]
            endDate = date[1]
            import random
            priceHist.append([item, shop, random.randrange(
                int(0.8*int(price)), int(1.2*int(price))), startDate, endDate])
        priceHist.append([item, shop, int(price), datePairs[len(
            datePairs)-1][0], datePairs[len(datePairs)-1][1]])
        break

for i in priceHist[0:12]:
    print(i)
    with open("price_history.csv", "w", newline="") as f:
     writer = csv.writer(f)
     writer.writerows(priceHist)

```

Contributions

Name	Individual Contribution to Submission (Lab 5)	Percentage of Contribution	Sign
Arora Kanupriya	Report, Table creation, Table population, Queries	16.67%	[illegible]
Malavade Sanskar Deepak	Table creation, population , queries and report	16.67%	[illegible]
Parashar Kshitij	Data creation, Table population, Queries	16.67%	[illegible]
Arnav Jaiswal	Report, Table creation, Table population, Queries	16.67%	[illegible]
Dhanyamraju Harsh Rao	Report, Table creation, Table population, Queries	16.67%	[illegible]
Neel Kumar	Report, Data creation, Table population, MP4 recordings	16.67%	Neel