

# מיני פרויקט בבסיסי נתונים

מגישות: טלי פרלא 325324713

[Taliaperlla@gmail.com](mailto:Taliaperlla@gmail.com)

ודבי רוזנברג 214378218 [debiroz10@gmail.com](mailto:debiroz10@gmail.com)

תשפ"ד

## תוכן העניינים:

### שלב 1

2	תיאור הארגון
3	ERD
3	הסבר כללי
3	DSD
4	CREAT TABLE
5	DROP TABEL
7	INSERT
8	TEXT
11	DATAGENERTOR
13	SELECTALL
14	גיבוי ושחזור

### שלב 2

16	שאלות SELECT
20	שאלות עדכון
22	שאלות מחיקה
24	שאלות פרמטרים
28	אילוצים
30	גיבוי

### שלב 3

32	תוכנית 1
32	פרוצדורה
34	פונקציה
36	תוכנית ראשית
39	תוכנית 2
39	פרוצדורה
40	פונקציה
41	תוכנית ראשית
43	גיבוי

### שלב 4

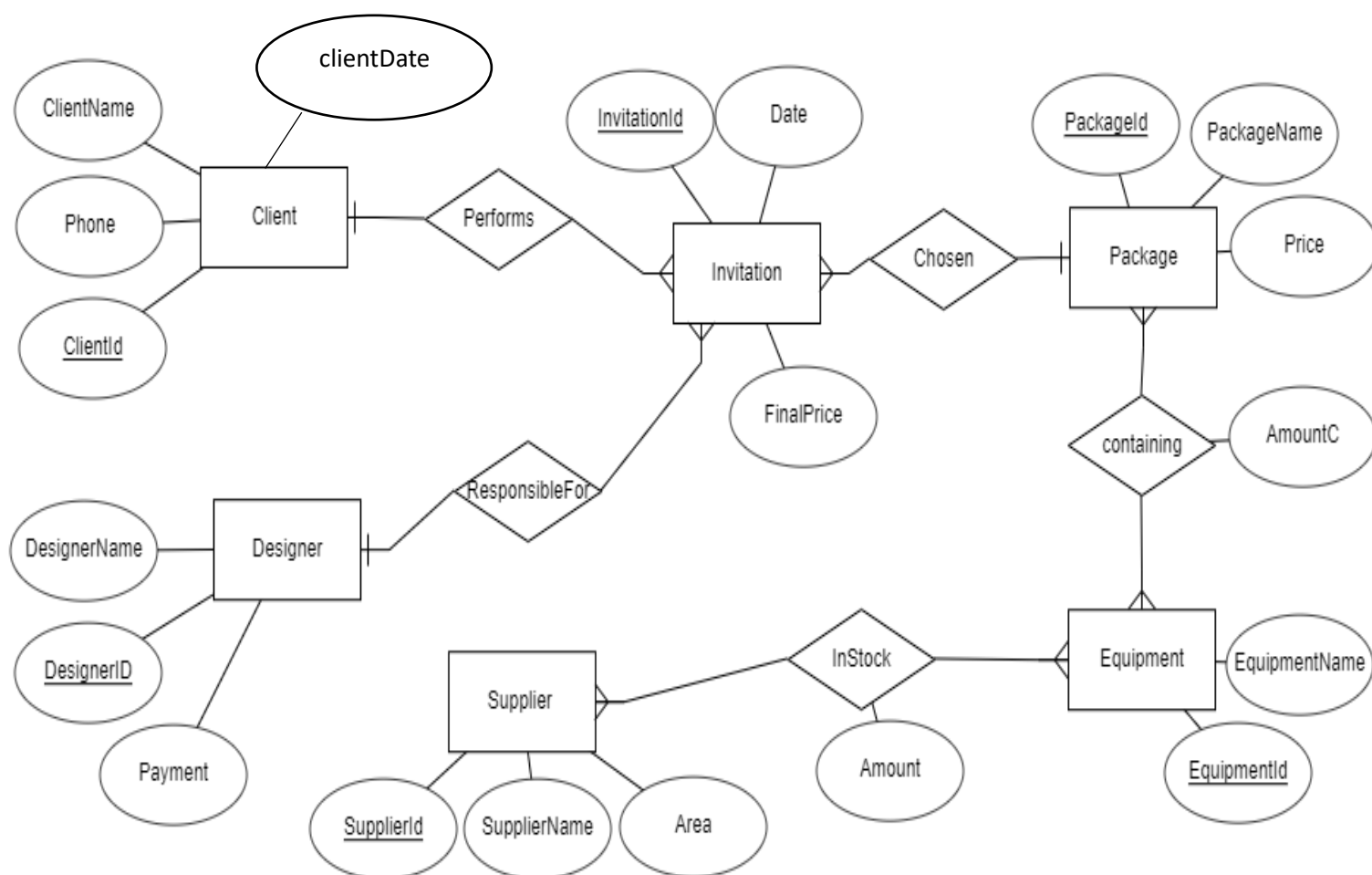
44	אינטגרציה
44	ERD ו DSD משותף
46	פקודות עיצוב
49	ERD ו DSD משותף
50	VIEWS

## תיאור הארגון:

### ארגון עיצוב פרחים לאירועים

הארגון נותן שירותי עיצוב לאירועים שונים כמו חתונות, בריתות, אירוסין, ימי הולדת, אירועי השקה, כנסים ועוד לקוחות מבצעים הזמנה ובכך יוצרים קשר עם מעצב ובוחרים חבילה המתאימה להם מהקטלוג. הקטלוג מכיל חבילות אופציונליות להזמנה וקשור לכמויות המלאי של הספקים מה שמאפשר לוודא שכל הפריטים בחבילה נמצאים במלאי ואפשר לבצע את ההזמנה. המערכת מאפשרת להזמין חבילה ממגוון רחב של ספקים על מנת להגיע לניצול המקסימלי של המשאבים.

### תרשים ERD:



### הסבר כללי:

חבילה - בחבילה יש פירוט על המוצרים המוצעים-שם, מחיר, מספר זיהוי מקושרת בקשר של "מכיל" לצידוד כלומר כל חבילה מכילה את הצידוד הרצוי בכמות מסוימת. בנוסף מחוברת בקשר "נבחר ב" של בדיוק אחד להזמנה כך שלכל הזמנה יש בדיוק חבילה אחת.

ציוד - מחובר בקשר של רבים לרבים לחבילה וספק. כך שכל חבילה מכילה כמות מסוימת של ציוד וכל ספק מוודא שהכמות הרצויה של הציוד הנדרש קיימת במלאי. מפורט בו שם ומספר מזהה.

ספק - מחבר בקשר "קיים במלאי" של רבים לרבים לישות "ציוד" - כל ספק מוודא שהציוד נמצא במלאי בכמות הרצויה. מפורט בו שם ספק, מספר מזהה, אזור הספקה.

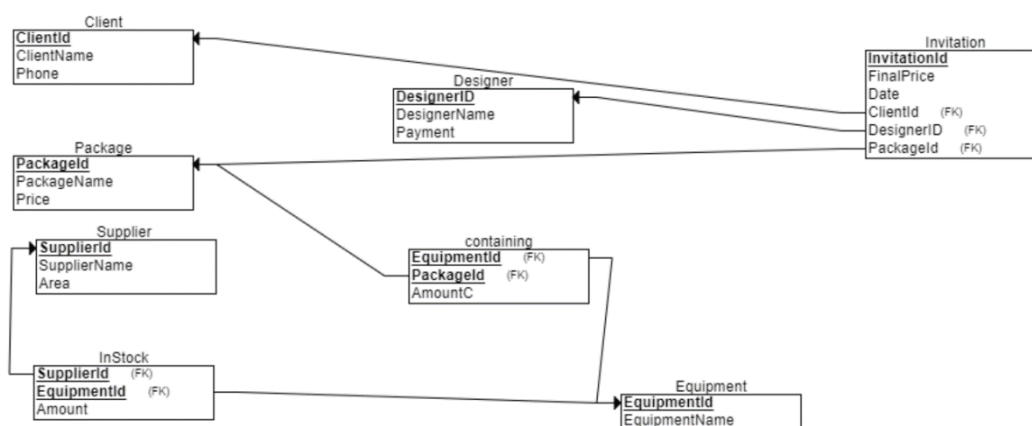
הזמנה - מחוברת בקשר של בדיוק אחד לישויות: לקוח, מעצב, וחבילה - מוגבלת כל שבכל הזמנה יש לקוח אחד, חבילה אחת, ומעצב אחד. מפורט בה תאריך, מספר הזמנה, ומחיר סופי.

לקוח - מפורט בו שם מלא, טלפון, ות"ז מחובר בקשר של בדיוק אחד "מבצע את" כל שלכל הזמנה יש לקוח אחד שמבצע אותה.

מעצב - מפורט בו שם המעצב, ת"ז, תשלום למעצב. מחובר בקשר של בדיוק אחד "אחראי לביצוע" להזמנה כך שלכל הזמנה יש מעצב אחד.

## שלב 1

### דיאגרמת DSD



### הטבלאות שנוצרו:

לקוח (מספר לקוח, שם לקוח, תאריך, טלפון)  
 ספק (מספר ספק, שם ספק, אזור)  
 חבילה (מספר חבילה, שם חבילה, מחיר)  
 מעצב (מספר מעצב, שם מעצב, תשלום)  
 ציוד (מספר ציוד, שם ציוד)  
 הזמנה (מספר הזמנה, מחיר סופי, תאריך, מספר לקוח, מספר מעצב, מספר חבילה)  
 במלאי (כמות, מספר מעצב, מספר ציוד)  
 מכיל (מספר חבילה, מספר ציוד, כמות)

### יצירה טבלאות ע"י אורקל

```

CREATE TABLE Client
(
  ClientId VARCHAR(8) NOT NULL,
  ClientName VARCHAR(15) NOT NULL,

```

```

    Phone VARCHAR(10) NOT NULL,
    BirthDate DATE NOT NULL,
    PRIMARY KEY (ClientId)
);

CREATE TABLE Supplier_
(
    SupplierId NUMERIC(5) NOT NULL,
    SupplierName VARCHAR(15) NOT NULL,
    Area VARCHAR(30),
    PRIMARY KEY (SupplierId)
);

CREATE TABLE Pakcage
(
    PackageId NUMERIC(5) NOT NULL,
    PackageName VARCHAR(15) NOT NULL,
    Price NUMERIC(5) NOT NULL,
    PRIMARY KEY (PackageId)
);

CREATE TABLE Designer
(
    DesignerName VARCHAR(15) NOT NULL,
    DesignerID NUMERIC(5) NOT NULL,
    Payment NUMERIC(5) NOT NULL,
    PRIMARY KEY (DesignerID)
);

CREATE TABLE Equipment
(
    EquipmentId NUMERIC(5) NOT NULL,
    EquipmentName VARCHAR(15) NOT NULL,
    PRIMARY KEY (EquipmentId)
);

CREATE TABLE InStock
(
    Amount NUMERIC(5) NOT NULL,
    SupplierId NUMERIC(5) NOT NULL,
    EquipmentId NUMERIC(5) NOT NULL,
    PRIMARY KEY (SupplierId, EquipmentId),
    FOREIGN KEY (SupplierId) REFERENCES Supplier_(SupplierId),
    FOREIGN KEY (EquipmentId) REFERENCES Equipment(EquipmentId)
);

CREATE TABLE containing
(
    AmountC NUMERIC(5) NOT NULL,
    EquipmentId NUMERIC(5) NOT NULL,
    PackageId NUMERIC(5) NOT NULL,
    PRIMARY KEY (EquipmentId, PackageId),
    FOREIGN KEY (EquipmentId) REFERENCES Equipment(EquipmentId),
    FOREIGN KEY (PackageId) REFERENCES Pakcage(PackageId)
);

CREATE TABLE Invitation
(
    FinalPrice NUMERIC(7),
    InvitationId NUMERIC(5) NOT NULL,
    Datte DATE NOT NULL,

```

```

    ClientId VARCHAR(8) NOT NULL,
    DesignerId NUMERIC(5) NOT NULL,
    PackageId NUMERIC(5) NOT NULL,
    PRIMARY KEY (InvitationId),
    FOREIGN KEY (ClientId) REFERENCES Client (ClientId),
    FOREIGN KEY (DesignerID) REFERENCES Designer (DesignerID),
    FOREIGN KEY (PackageId) REFERENCES Pakcage (PackageId)
);

```

### מחיקה:

```

drop table InStock;
drop table containing;
drop table Invitation;
drop table Client;
drop table Supplier_;
drop table Pakcage;
drop table Designer;
drop table Equipment;

```

### הצגת מבנה טבלה עי ORACLE

```

Connected to Oracle Database 11g Express Edition Release 11.2.0.2.0
Connected as debbil@XE

```

```

SQL> desc CLIENT
Name          Type          Nullable Default Comments
-----
CLIENTID      VARCHAR2(9)
CLIENTNAME    VARCHAR2(15)
PHONE         VARCHAR2(10)

```

```

SQL> desc SUPPLIER_
Name          Type          Nullable Default Comments
-----
SUPPLIERID     NUMBER(5)
SUPPLIERNAME   VARCHAR2(15)
AREA          VARCHAR2(30) Y

```

```

SQL> desc PAKCAGE
Name          Type          Nullable Default Comments
-----
PACKAGEID     NUMBER(5)
PACKAGENAME   VARCHAR2(50)
PRICE        NUMBER(5)

```

```

SQL> desc DESIGNER
Name          Type          Nullable Default Comments
-----
DESIGNERNAME  VARCHAR2(15)
DESIGNERID    NUMBER(5)
PAYMENT       NUMBER(5)

```

```

SQL> desc EQUIPMENT
Name          Type          Nullable Default Comments
-----
EQUIPMENTID   NUMBER(5)
EQUIPMENTNAME VARCHAR2(50)

```

```

SQL> desc INSTOCK

```

Name	Type	Nullable	Default	Comments
AMOUNT	NUMBER(5)			
SUPPLIERID	NUMBER(5)			
EQUIPMENTID	NUMBER(5)			

SQL> desc CONTAINING

Name	Type	Nullable	Default	Comments
AMOUNTC	NUMBER(5)			
EQUIPMENTID	NUMBER(5)			
PACKAGEID	NUMBER(5)			

SQL> desc INVITATION

Name	Type	Nullable	Default	Comments
FINALPRICE	NUMBER(7)	Y		
INVITATIONID	NUMBER(5)			
DATTE	DATE			
CLIENTID	VARCHAR2(8)			
DESIGNERID	NUMBER(5)			
PACKAGEID	NUMBER(5)			

## הכנסת נתונים לטבלאות:

הכנסה ידנית **INSERT**

```
select * from CLIENT;
insert into CLIENT (CLIENTID, CLIENTNAME, PHONE)
values (21437821, 'debi', 0547683445);
insert into CLIENT (CLIENTID, CLIENTNAME, PHONE)
values (21837821, 'dani', 0548983445);
insert into CLIENT (CLIENTID, CLIENTNAME, PHONE)
values (56787821, 'tali', 0547683445);
insert into CLIENT (CLIENTID, CLIENTNAME, PHONE)
values (27837821, 'yair', 0523478945);
insert into CLIENT (CLIENTID, CLIENTNAME, PHONE)
values (214389221, 'chana', 039090555);
insert into CLIENT (CLIENTID, CLIENTNAME, PHONE)
values (21482021, 'dadi', 0537829887);
insert into CLIENT (CLIENTID, CLIENTNAME, PHONE)
values (34567821, 'mali', 0533456778);
insert into CLIENT (CLIENTID, CLIENTNAME, PHONE)
values (02892028, 'noa', 0537862116);
insert into CLIENT (CLIENTID, CLIENTNAME, PHONE)
values (28208921, 'moshe', 054098765);
insert into CLIENT (CLIENTID, CLIENTNAME, PHONE)
values (345678291, 'orit', 0543283445);
COMMIT;
```

	CLIENTID	CLIENTNAME	PHONE
1	214378218	debi	547683445
2	214378214	dani	548983445
3	567878214	tali	547683445
4	278378218	yair	523478945
5	214820218	dadi	537829887
6	345678218	mali	533456778
7	28920218	noa	537862116
8	282089218	moshe	54098765
9	21437821	debi	547683445
10	56787821	tali	547683445
11	27837821	yair	523478945
12	214389221	chana	39090555
13	21482021	dadi	537829887
14	34567821	mali	533456778
15	2892028	noa	537862116
16	28208921	moshe	54098765
17	345678291	orit	543283445
18	21837821	dani	548983445



SQL Output Statistics

```

select * from SUPPLIER;
insert into SUPPLIER_ (SUPPLIERID,SUPPLIERNAME,AREA)
values(12345,'debi','north');
insert into SUPPLIER_ (SUPPLIERID,SUPPLIERNAME,AREA)
values(12365,'chani','south');
insert into SUPPLIER_ (SUPPLIERID,SUPPLIERNAME,AREA)
values(14565,'yael','south');
insert into SUPPLIER_ (SUPPLIERID,SUPPLIERNAME,AREA)
values(89065,'meir','gushdDan');
insert into SUPPLIER_ (SUPPLIERID,SUPPLIERNAME,AREA)
values(12395,'chavi','jerusalem');
insert into SUPPLIER_ (SUPPLIERID,SUPPLIERNAME,AREA)
values(87655,'dasi','gushdDan');
insert into SUPPLIER_ (SUPPLIERID,SUPPLIERNAME,AREA)
values(89365,'riki','north');
insert into SUPPLIER_ (SUPPLIERID,SUPPLIERNAME,AREA)
values(23565,'eti','south');
insert into SUPPLIER_ (SUPPLIERID,SUPPLIERNAME,AREA)
values(19475,'talía','jerusalem');
insert into SUPPLIER_ (SUPPLIERID,SUPPLIERNAME,AREA)
values(15365,'noa','gushdDan');

```

	SUPPLIERID	SUPPLIERNAME	AREA
1	12345	debi	north
2	12365	chani	south
3	14565	yael	south
4	89065	meir	gushdDan
5	12395	chavi	jerusalem
6	87655	dasi	gushdDan
7	89365	riki	north
8	23565	eti	south
9	19475	talía	jerusalem
10	15365	noa	gushdDan

## TEXT - יצרנו קובץ טקסט עם הנתונים וייבאנו לתוכנה

Data from Textfile Data to Oracle

General

Owner: DESIGNER

Table: DESIGNER

Clear Table

Commit every: 0

Ignore duplicates

Fields

Field1 -> DESIGNERID (NUMBER)

Field2 -> DESIGNERNAME (VARCHAR2)

Field3 -> PAYMENT (NUMBER)

Result Preview

1	2	3
1	Debbie	5000
2	Daniel	8000

Import Import to Script Close debbi@XE DESIGNER.txt lo

Data from Textfile Data to Oracle

File Data

```

1, Debbie, 5000
2, Daniel, 8000
3, Anna, 4500
4, John, 7000
5, Linda, 6500
6, Michael, 9200
7, Emily, 4800
8, Robert, 7600
9, Laura, 5300
10, David, 8400
11, Jessica, 4900
12, Richard, 7100
13, Karen, 6500
14, Thomas, 7500
15, Susan, 5700
16, Charles, 8600
17, Sarah, 8700
18, Joseph, 7300
19, Karen, 6900
20, James, 8300
21, Patricia, 6000
22, Christopher, 9100
23, Barbara, 8800
24, Matthew, 7800
25, Elizabeth, 6100

```

Configuration

General

Fieldcount: 3

Quote character: "

Field1 (+0..?)

Field2 (+0..?)

Field3 (+0..?)

Field Start

Relative position

Absolute position

Character

Field End

Length

Character

Filter

Result Preview

1	2	3
1	Debbie	5000
2	Daniel	8000

Import Import to Script Close debbi@XE DESIGNER.txt loaded, 3 KB

בדיקה שעובד:

עמוד 9 מתוך 54

Data from Textfile Data to Oracle

General

Owner Table ☐ Clear Table

Commit every... ☐ Overwrite duplicates ☒ Ignore duplicates

Fields

Field1 -> SUPPLIERID (NUMBER)  
Field2 -> SUPPLIERNAME (VARCHAR2)  
Field3 -> AREA (VARCHAR2)

Result Preview

1	2	3
1	bridachair	4000
2	Tabledecoration	5000

Import Import to Script Close debbi1@XE Package.txt loaded, 1 KB

Data from Textfile Data to Oracle

File Data

1, bridachair,4000  
2, Tabledecoration, 5000  
3, Bridalbouquet, 2000  
4, Bridalchair + tabledesign, 7000  
5,Backtoabridalchair ,5000  
6, Bardesign, 6000  
7, tablegarlands, 7000  
8, Backtoabridalchair+bridachair, 7000  
9, tablegarlands+Tabledecoration, 8000  
10, Designedchandeliers, 10000

Configuration

General

Fieldcount 3

Quote character " " " "

Field1 (+0..")  
Field2 (+0..")  
Field3 (+0..")

Field Start  
☐ Relative position  
☐ Absolute position  
☐ Character

Field End  
☐ Length  
☐ Character

Filter

Result Preview

1	2	3
1	bridachair	4000
2	Tabledecoration	5000

Import Import to Script Close debbi1@XE Package.txt loaded, 1 KB

## בדיקה שעובד:

```
select * from DESIGNER;
select * from PACKAGE;
```

	PACKAGEID	PACKAGENAME	PRICE
1	1	bridachair	4000
2	2	Tabledecoration	5000
3	3	Bridalbouquet	2000
4	4	Bridalchair + tabledesign	7000
5	6	Bardesign	6000
6	7	tablegarlands	7000
7	8	Backtoabridalchair+bridachair	7000
8	9	tablegarlands+Tabledecoration	8000
9	10	Designedchandeliers	10000

## DATA Generation

INSTOCK

Owner	Table	Number of records
DEBBI1	INSTOCK	200

Name	Type	Size	Data
AMOUNT	NUMBER	5	Random(10, 99999)
SUPPLIERID	NUMBER	5	List(select SUPPLIERID from SUPPLIER_)
EQUIPMENTID	NUMBER	5	Sequence(1)
*			

INSTOCK

AMOUNT	SUPPLIERID	EQUIPMENTID
53069	89065	162
79044	15365	163
51838	23565	164
46962	89365	165
8917	12395	166
49125	23565	167
18204	12345	168
87842	15365	169
3425	12395	170
5259	89365	171
47027	14565	172
93906	12365	173
43060	89065	174
5569	87655	175
3613	14565	176
255	14565	177
73579	87655	178
2896	89065	179
33450	87655	180
59618	12345	181
94673	15365	182

```

insert into DEBB11.INSTOCK (AMOUNT, SUPPLIERID, EQUIPMENTID)
values (70606, 14565, 1);

insert into DEBB11.INSTOCK (AMOUNT, SUPPLIERID, EQUIPMENTID)
values (58443, 23565, 2);

insert into DEBB11.INSTOCK (AMOUNT, SUPPLIERID, EQUIPMENTID)
values (6161, 89365, 3);

insert into DEBB11.INSTOCK (AMOUNT, SUPPLIERID, EQUIPMENTID)
values (80676, 89365, 4);

insert into DEBB11.INSTOCK (AMOUNT, SUPPLIERID, EQUIPMENTID)
values (58559, 14565, 5);

insert into DEBB11.INSTOCK (AMOUNT, SUPPLIERID, EQUIPMENTID)
values (87826, 19475, 6);

insert into DEBB11.INSTOCK (AMOUNT, SUPPLIERID, EQUIPMENTID)

```

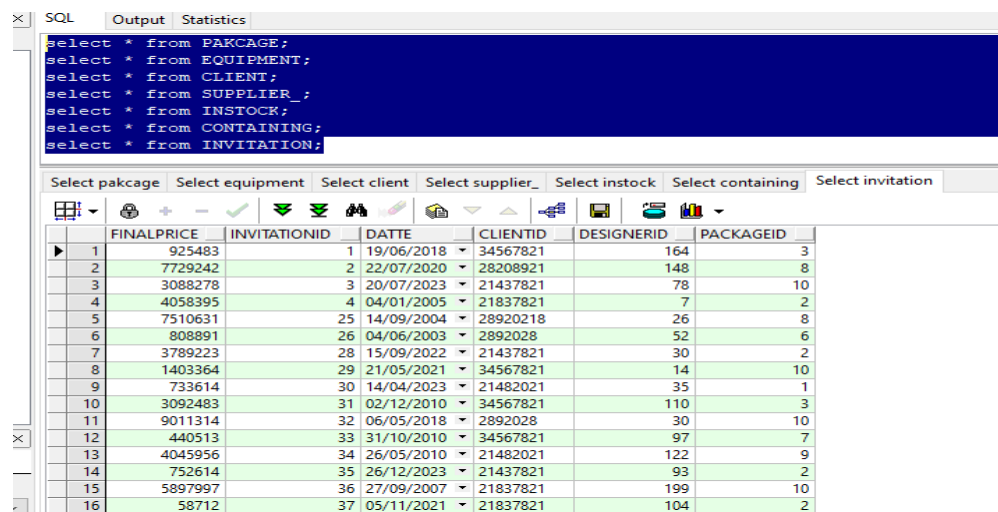
**Data Generator - New**

Owner: DEBB11    Table: INVITATION    Number of records: 200

Name	Type	Size	Data
FINALPRICE	NUMBER	7	Random(100, 99999999)
INVITATIONID	NUMBER	5	Sequence(115)
DATTE	DATE		Random(1/1/2000, 1/1/2027)
CLIENTID	VARCHAR2	8	List(select CLIENTID from CLIENT)
DESIGNERID	NUMBER	5	List(select DESIGNERID from DESIGNER)
PACKAGEID	NUMBER	5	List(select PACKAGEID from PAKCAGE)
*			

לאחר שסימנו למלאות את הטבלאות בכל אחת מהשיטות בדקנו שהכל התמלאה נכון וטוב.

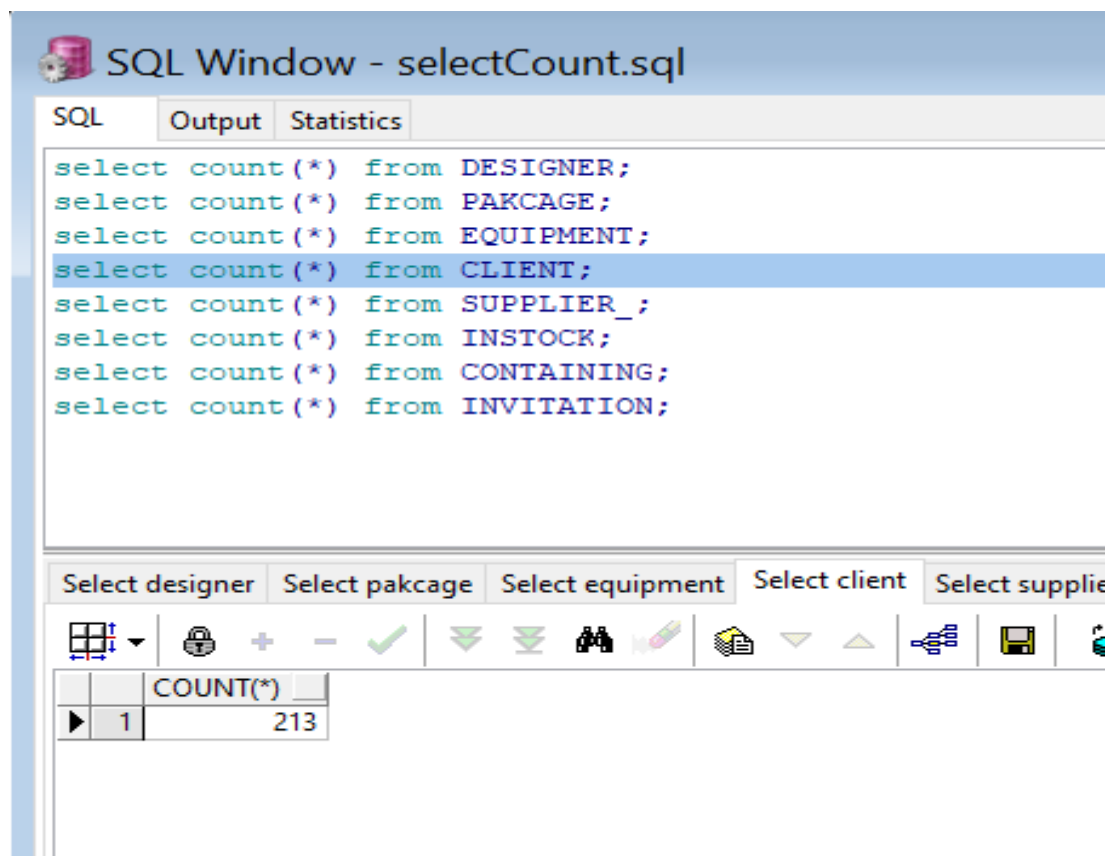
## פקודת SELECT



The screenshot shows the SQL Enterprise Manager interface. The SQL pane contains a query that selects all columns from seven tables: PACKAGE, EQUIPMENT, CLIENT, SUPPLIER, INSTOCK, CONTAINING, and INVITATION. The Output pane displays the results of this query as a table with 16 rows and 7 columns: FINALPRICE, INVITATIONID, DATTE, CLIENTID, DESIGNERID, and PACKAGEID. The results are sorted by INVITATIONID in ascending order.

	FINALPRICE	INVITATIONID	DATTE	CLIENTID	DESIGNERID	PACKAGEID
1	925483	1	19/06/2018	34567821	164	3
2	7729242	2	22/07/2020	28208921	148	8
3	3088278	3	20/07/2023	21437821	78	10
4	4058395	4	04/01/2005	21837821	7	2
5	7510631	25	14/09/2004	28920218	26	8
6	808891	26	04/06/2003	2892028	52	6
7	3789223	28	15/09/2022	21437821	30	2
8	1403364	29	21/05/2021	34567821	14	10
9	733614	30	14/04/2023	21482021	35	1
10	3092483	31	02/12/2010	34567821	110	3
11	9011314	32	06/05/2018	2892028	30	10
12	440513	33	31/10/2010	34567821	97	7
13	4045956	34	26/05/2010	21482021	122	9
14	752614	35	26/12/2023	21437821	93	2
15	5897997	36	27/09/2007	21837821	199	10
16	58712	37	05/11/2021	21837821	104	2

## פקודת SELECT COUNT

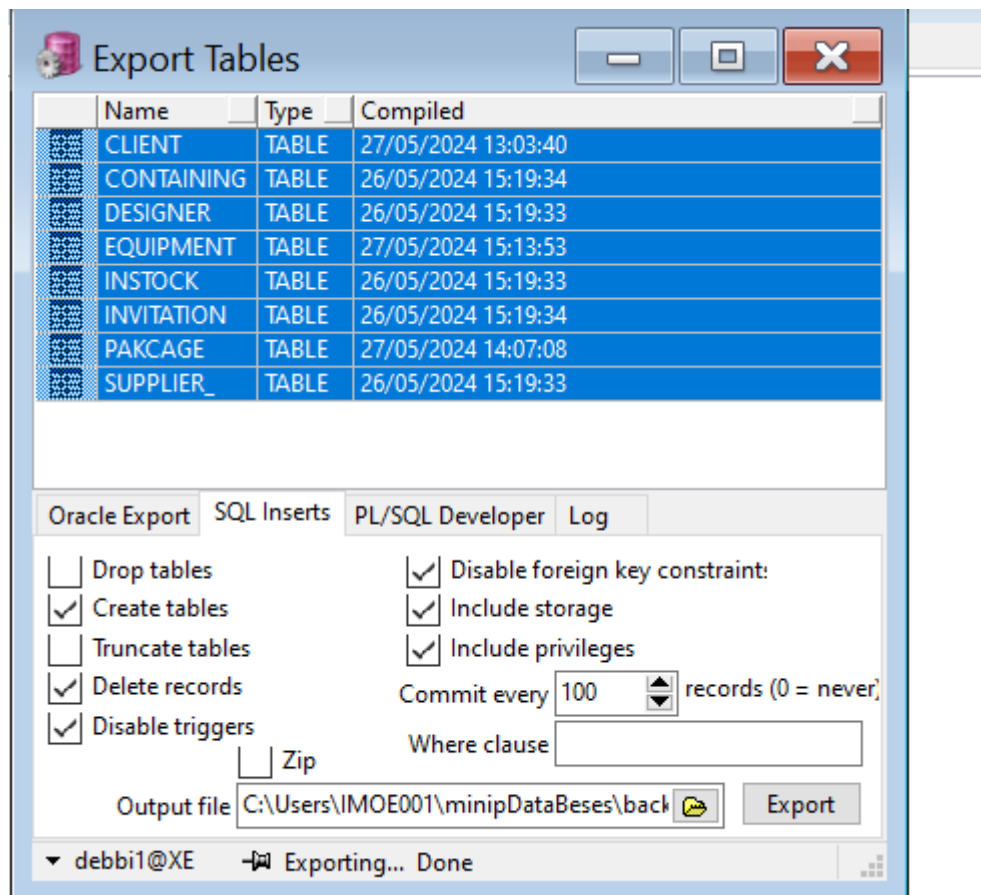


The screenshot shows the SQL Enterprise Manager interface. The SQL pane contains a query that uses the COUNT(\*) function to count the number of rows in seven tables: DESIGNER, PACKAGE, EQUIPMENT, CLIENT, SUPPLIER, INSTOCK, and CONTAINING. The Output pane displays the results of this query as a table with 1 row and 2 columns: COUNT(\*) and 213.

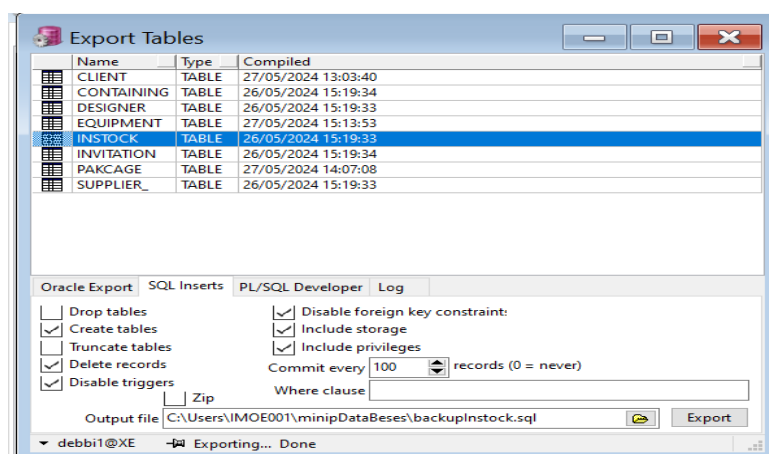
	COUNT(*)
1	213

## גיבוי

Oracle Export נכנסתי ללשונית tools ובחרתי export tables בחרתי את הלשונית sql insert והכנסתי את כתובת היעד בה יישמר הגיבוי. לאחר לחיצה על export נשאלתי האם ברצוני לגבות את כל הטבלאות ונוצר קובץ גיבוי ובו נתוני כל הטבלאות.



באותו אופן יצרנו גיבוי לכל אחת מהטבלאות בנפרד



בס"ד

## שחזור

לאחר יצירת הגיבוי, רציתי לוודא את אמינותו ומחקתי את תוכן טבלת הלקוחות. כפי שניתן לראות, כעת טבלת הלקוחות ריקה. שוב נכנסתי ל tools והפעם בחרתי import tables כקובץ מקור את קובץ הגיבוי לטבלת הלקוחות ולחצתי על import כעת, טבלת הלקוחות נטענה שוב ב 215 הרשומות שמחקנו.

SQL Window - selectTables.sql

SQL Output Statistics

```
select * from DESIGNER;  
select * from PACKAGE;  
select * from EQUIPMENT;  
select * from CLIENT;  
select * from SUPPLIER_  
select * from INSTOCK;  
select * from CONTAINING;  
select * from INVITATION;
```

	CLIENTID	CLIENTNAME	PHONE
203	209	Goldie	9742647628
204	210	Machine	7732987579
205	211	Juliet	9486741337
206	212	Willem	1327728799
207	213	Pat	5272384725
208	214	Luke	4992734354
209	215	Connie	3416565562
210	216	Clarence	5125468112
211	217	Harris	8573822188
212	218	Geoff	6135139693
213	219	Randy	1515877945

& 4:1 debbi1@XE 213 rows selected in 0.218 seconds

SQL Window - selectTables.sql

SQL Output Statistics

```
select * from DESIGNER;  
select * from PACKAGE;  
select * from EQUIPMENT;  
select * from CLIENT;  
select * from SUPPLIER_  
select * from INSTOCK;  
select * from CONTAINING;  
select * from INVITATION;
```

	CLIENTID	CLIENTNAME	PHONE
--	----------	------------	-------



## שלב 2

### שאלות SELECT

הסבר:

השאלתה מחזירה את הספק שממנו הזמינו הכי הרבה ציוד

The screenshot shows a SQL query window with the following query:

```
SELECT s.SupplierName, COUNT(i.InvitationId) AS total_orders
FROM Supplier_ s
JOIN InStock ins ON s.SupplierId = ins.SupplierId
JOIN Equipment e ON ins.EquipmentId = e.EquipmentId
JOIN containing c ON e.EquipmentId = c.EquipmentId
JOIN Package p ON c.PackageId = p.PackageId
JOIN Invitation i ON p.PackageId = i.PackageId
GROUP BY s.SupplierName
HAVING COUNT(i.InvitationId) = (
    SELECT MAX(total_orders)
    FROM (
        SELECT COUNT(i2.InvitationId) AS total_orders
        FROM Supplier_ s2
        JOIN InStock is2 ON s2.SupplierId = is2.SupplierId
        JOIN Equipment e2 ON is2.EquipmentId = e2.EquipmentId
        JOIN containing c2 ON e2.EquipmentId = c2.EquipmentId
        JOIN Package p2 ON c2.PackageId = p2.PackageId
        JOIN Invitation i2 ON p2.PackageId = i2.PackageId
        GROUP BY s2.SupplierName
    ) max_orders
);
```

The results pane shows a single row:

	SUPPLIERNAME	TOTAL_ORDERS
1	chani	225

הסבר:

שאלתה למציאת המחיר הממוצע של חבילות שהוזמנו על ידי לקוחות שהשם שלהם מתחיל במ

The screenshot shows a SQL query window with the following query:

```
SELECT AVG(i.FinalPrice) AS average_price
FROM Invitation i
JOIN Client c ON i.ClientId = c.ClientId
JOIN Package p ON i.PackageId = p.PackageId
WHERE c.ClientName LIKE 'm%';
```

The results pane shows a single row:

	AVERAGE_PRICE
1	4234090.37142857

## הסבר

שאלתה זו מספקת תצוגה מקיפה של היסטוריית ההזמנות של כל לקוח, כולל מספר ההזמנות שבוצעו ותאריך ההזמנה האחרונה שלו, ומזמנת את התוצאות כך שהלקוחות עם הכי הרבה הזמנות יופיעו ראשונים.

9 rows selected in 0.063 seconds

SQL Output Statistics

```

SELECT
  c.ClientName,
  COUNT(i.InvitationId) AS TotalOrders,
  MAX(i.Date) AS LastOrderDate
FROM
  Client c
JOIN
  Invitation i ON c.ClientId = i.ClientId
JOIN
  Designer d ON i.DesignerId = d.DesignerID
GROUP BY
  c.ClientId, c.ClientName
ORDER BY
  TotalOrders DESC;

```

	CLIENTNAME	TOTALORDERS	LASTORDERDATE
1	yair	29	18/08/2026
2	debi	27	29/05/2025
3	dadi	24	05/08/2025
4	tali	22	28/11/2026
5	mali	19	25/03/2022
6	moshe	16	20/06/2025
7	dani	14	08/01/2025
8	noa	14	23/10/2026
9	noa	11	14/02/2023

שמות הלקוחות הזמינו מעל 40000

SQL Output Statistics

```
SELECT DISTINCT c.ClientName  
FROM Invitation i  
JOIN Client c ON i.ClientId = c.ClientId  
WHERE i.FinalPrice > 40000;
```

CLIENTNAME

1	debi
2	dadi
3	mali
4	yair
5	dani
6	noa
7	tali
8	moshe

מחזיר את הלקוחות שסכום ההזמנות שלו מעל 100000 וגם 3 מתוכן עם אותה חבילה

SQL Output Statistics

```

WITH ClientOrderSummary AS (
    SELECT
        i.ClientId,
        SUM(i.FinalPrice) AS TotalAmount,
        COUNT(DISTINCT i.PackageId) AS PackageCount,
        COUNT(*) - COUNT(DISTINCT i.PackageId) AS SamePackageOrders
    FROM Invitation i
    GROUP BY i.ClientId
)
SELECT
    c.ClientId,
    c.ClientName
FROM
    ClientOrderSummary cos
JOIN
    Client c ON cos.ClientId = c.ClientId
WHERE
    cos.TotalAmount > 100000 AND cos.SamePackageOrders >= 3
ORDER BY
    cos.TotalAmount DESC;

```

CLIENTID CLIENTNAME

1	21482021	dadi
2	21437821	debi
3	27837821	yair
4	56787821	tali
5	34567821	mali
6	2892028	noa
7	28208921	moshe
8	21837821	dani
9	28920218	noa

## שאלות עדכון

הסבר

שאלת עדכון – המעצב עם תז  
17 העלה את המחיר ב15 אחוז

עקב קשיים ועלית מחירי  
הפרחים.

```
UPDATE Invitation  
SET FinalPrice = FinalPrice * 1.15  
WHERE DesignerID = 17;  
select * from Invitation;
```

לפני

170	3526891	214	29/04/2004	21437821	182	1
171	2083	215	16/05/2020	27837821	17	6
172	89900	217	16/07/2020	27837821	12	2
173	70050	218	16/06/2020	27837821	20	7
174	50000	219	17/04/2020	27837821	161	7
175	90000	220	14/05/2020	27837821	141	8
176	80000	221	16/08/2020	27837821	26	7
177	2347	229	13/05/2020	27837821	17	7

אחרי

169	1210428	213	05/05/2020	2892028	159	3
170	3526891	214	29/04/2004	21437821	182	1
171	2395	215	16/05/2020	27837821	17	6
172	89900	217	16/07/2020	27837821	12	2
173	70050	218	16/06/2020	27837821	20	7
174	50000	219	17/04/2020	27837821	161	7
175	90000	220	14/05/2020	27837821	141	8
176	80000	221	16/08/2020	27837821	26	7
177	2699	229	13/05/2020	27837821	17	7

הסבר

שאלתה שבודקת מי החבילה שסופקה הכי הרבה

```
SELECT  
    p.PackageName,  
    COUNT(i.InvitationId) AS OrderCount  
FROM  
    Invitation i  
JOIN  
    Package p ON i.PackageId = p.PackageId  
GROUP BY  
    p.PackageName  
ORDER BY  
    OrderCount DESC;
```

הסבר:

מדיניות החברה קבעה שחבילה שהוזמנה יותר מ-24 פעמים יעלה מחירה ב-1500 שקלים

```
-- Update the price of packages that have been ordered more than 24 times
UPDATE Package
SET Price = Price + 1500
WHERE PackageId IN (
    SELECT PackageId
    FROM Invitation
    GROUP BY PackageId
    HAVING COUNT(InvitationId) > 24
);
select * from PACKAGE;
```

לפני

	PACKAGEID	PACKAGENAME	PRICE
1	1	bridachair	4000
2	2	Tabledecoration	5000
3	3	Bridalbouquet	2000
4	4	Bridalchair + tabledesign	7000
5	6	Bardesign	6000
6	7	tablegarlands	7000
7	8	Backtoabridalchair+bridachair	7000
8	9	tablegarlands+ Tabledecoration	8000
9	10	Designedchandeliers	10000

אחרי

	PACKAGEID	PACKAGENAME	PRICE
1	1	bridachair	5500
2	2	Tabledecoration	5000
3	3	Bridalbouquet	2000
4	4	Bridalchair + tabledesign	7000
5	6	Bardesign	6000
6	7	tablegarlands	8500
7	8	Backtoabridalchair+bridachair	7000
8	9	tablegarlands+ Tabledecoration	8000
9	10	Designedchandeliers	10000

## שאלות מחיקה













שאלת שנותנת את כל הזמנות מ-2006

SQL

Output

Statistics

```
SELECT *
FROM Invitation
WHERE Datte >=to_date('01/01/2006','dd/mm/yyyy') AND Datte < to_date('01/01/2007','dd/mm/yyyy');
```

	FINALPRICE	INVITATIONID	DATTE	CLIENTID	DESIGNERID	PACKAGEID
▶ 1	9750932	43	08/11/2006 ▾	21437821	98	6
2	5776229	59	24/10/2006 ▾	56787821	119	1
3	2299977	72	27/12/2006 ▾	21437821	57	10
4	2173722	91	14/01/2006 ▾	21437821	72	4
5	526171	141	17/09/2006 ▾	21437821	97	7
6	205717	207	27/01/2006 ▾	27837821	147	1

הסבר-

עקב תקלה במערכת נוצרו הזמנות שגויות ללקוח שבחר בחבילה מספר 1

ונוצר כפל הזמנות סתם .

בשנת 2006 הוזמנה רק הזמנה אחת עם חבילה מספר 1 ונוצר טעות.(כפי שניתן לראות כי יש שתי הזמנות)

<pre>-- Delete unnecessary orders for package number 1 in 2006 DELETE FROM Invitation WHERE PackageId = 1 AND Datte &gt;= to_date('01/01/2006','dd/mm/yyyy') AND Datte &lt; to_date('01/01/2007','dd/mm/yyyy') AND InvitationId NOT IN (   SELECT MIN(InvitationId)   FROM Invitation   WHERE PackageId = 1 AND Datte &gt;= to_date('01/01/2006','dd/mm/yyyy') AND Datte &lt; to_date('01/01/2007','dd/mm/yyyy') ); SELECT *</pre>	
--	--

לאחר התיקון

```
SELECT *
FROM Invitation
WHERE Datte >=to_date('01/01/2006','dd/mm/yyyy') AND Datte < to_date('01/01/2007','dd/mm/yyyy');
```

	FINALPRICE	INVITATIONID	DATTE	CLIENTID	DESIGNERID	PACKAGEID
1	9750932	43	08/11/2006	21437821	98	6
2	5776229	59	24/10/2006	56787821	119	1
3	2299977	72	27/12/2006	21437821	57	10
4	2173722	91	14/01/2006	21437821	72	4
5	526171	141	17/09/2006	21437821	97	7

הסבר-

המעצב תומס שמספרו 13 חלה במחלת שחמת הכבד בצורה קשה מאד ונאלץ לעזוב את עבודתו ולקחת חופשת מחלה ממושכת

ולכן נסיר אותו מרשימת המעצבים העומדים לרשות החברה.

	DESIGNERNAME	DESIGNERID	PAYMENT
1	Debbie	1	5000
2	Daniel	2	8000
3	Anna	3	4500
4	John	4	7000
5	Linda	5	6200
6	Michael	6	9200
7	Emily	7	4800
8	Robert	8	7600
9	Laura	9	5300
10	David	10	8400
11	Jessica	11	4900
12	Richard	12	7100
13	Karen	13	6500
14	Thomas	14	7900
15	Susan	15	5700
16	Charles	16	8600
17	Sarah	17	4700
18	Joseph	18	7300
19	Karen	19	6900
20	James	20	8300
21	Patricia	21	6000
22	Christopher	22	9100
23	Barbara	23	5800
24	Matthew	24	7800
25	Elizabeth	25	6100
26	Anthony	26	8500
27	Lisa	27	4900

```
DELETE FROM Designer
WHERE DesignerID = 13;
```

```
select * from DESIGNER t
```

	DESIGNERNAME	DESIGNERID	PAYMENT
1	Debbie	1	5000
2	Daniel	2	8000
3	Anna	3	4500
4	John	4	7000
5	Linda	5	6200
6	Michael	6	9200
7	Emily	7	4800
8	Robert	8	7600
9	Laura	9	5300
10	David	10	8400
11	Jessica	11	4900
12	Richard	12	7100
13	Thomas	14	7900
14	Susan	15	5700
15	Charles	16	8600
16	Sarah	17	4700
17	Joseph	18	7300
18	Karen	19	6900
19	James	20	8300
20	Patricia	21	6000
21	Christopher	22	9100
22	Barbara	23	5800



## שאלות פרמטרים

הסבר- שאלתה שמאחזרת את כל הלקוחות שביצעו הזמנות עם סכום כולל הגבוה מערך שהמשתמש מקליד מתאריך עד תאריך מסוים עפ בחירת המשתמש.

The screenshot shows the SQL Developer interface. The SQL window contains the following query:

```
SELECT
  c.ClientName,
  SUM(i.FinalPrice) AS TotalSpent
FROM
  Client c
JOIN
  Invitation i ON c.ClientId = i.ClientId
WHERE
  i.Datte BETWEEN TO_DATE('<name="startDate" required="true">', 'dd/mm/yyyy') AND TO_DATE('<name="endDate" required="true">', 'dd/mm/yyyy')
GROUP BY
  c.ClientName
HAVING
  SUM(i.FinalPrice) > <name="minAmount" required="true">;
```

A Variables dialog box is open, showing the following parameters:

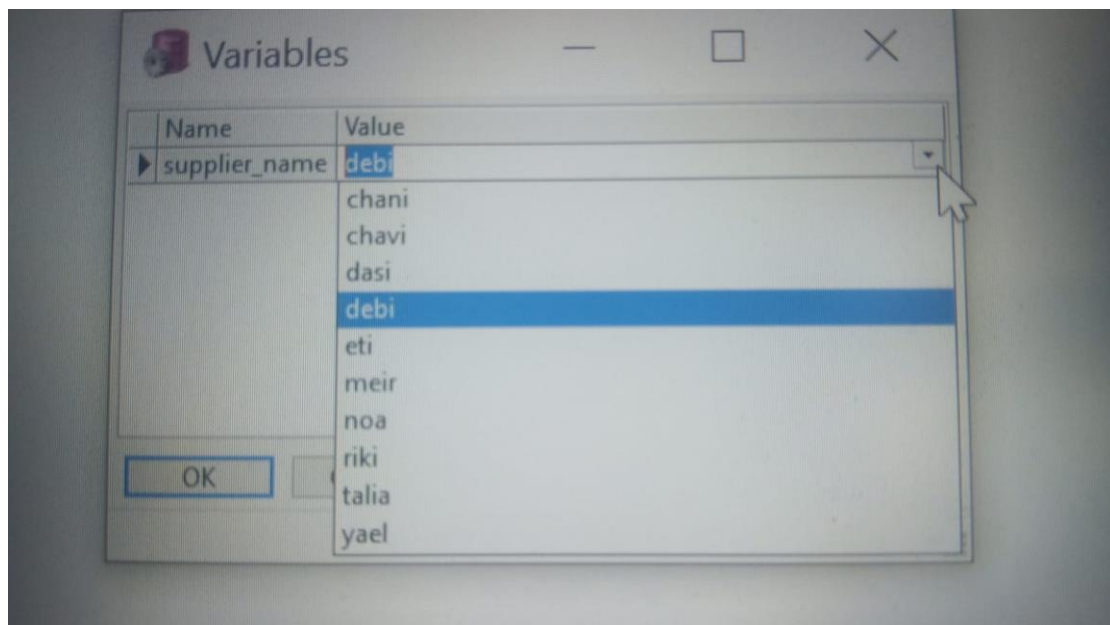
Name	Value
startDate	20/03/2023
endDate	27/12/2023
minAmount	10000

תוצאה-

	CLIENTNAME	TOTALSPENT
1	debi	10493766
2	dadi	10526382
3	yair	9450919
4	tali	3445522

הסבר- המשתמש יבחר מתוך רשימה או יזין את שם הספק מתוך רשימה, והשאלתה תחזיר את הציוד שסופק על ידי אותו ספק, בצירוף הכמויות במלאי.

```
SELECT
    s.SupplierName,
    e.EquipmentName,
    i.Amount
FROM
    Supplier_ s
JOIN
    InStock i ON s.SupplierId = i.SupplierId
JOIN
    Equipment e ON i.EquipmentId = e.EquipmentId
WHERE
    s.SupplierId = <name="supplier_name"
    list="SELECT SupplierId, SupplierName FROM Supplier_ ORDER BY SupplierName"
    description="yes" restricted="yes">;
```



תוצאה

	SUPPLIERNAME	EQUIPMENTNAME	AMOUNT
1	debi	Iris	96099
2	debi	Flower Pot	83587
3	debi	Weed Whacker	25219
4	debi	GardenSpade	5675
5	debi	thin vase	42500
6	debi	whiteraner	8887
7	debi	LilyoftheValley	75540

הסבר-

שאלתה זו מאחזרת את כל ההזמנות שבוצעו על ידי לקוחות שמתחילים מתאריך מסוים

SQL | Output | Statistics

```
SELECT
  c.ClientName,
  i.Datte,
  i.FinalPrice
FROM
  Invitation i
JOIN
  Client c ON i.ClientId = c.ClientId
WHERE
  i.Datte >= TO_DATE('&name="start_date" hint="Enter the start date (DD/MM/YYYY)">', 'DD/MM/YYYY')
ORDER BY
  i.Datte;
```

	CLIENTNAME	DATTE	FINALPRICE
1	mali	19/06/2018	925483
2	dani	17/07/2018	3861676
3	noa	04/11/2018	7750837
4	mali	24/12/2018	1655982
5	yair	25/12/2018	4482610
6	tali	18/01/2019	5012138
7	moshe	26/01/2019	2758243
8	yair	25/02/2019	4525396
9	dani	14/04/2019	4599656
10	dadi	24/04/2019	5057572
11	debi	08/05/2019	5598241
12	dadi	10/05/2019	8290798
13	mali	31/10/2019	6685067
14	mali	26/11/2019	1237383
15	tali	07/01/2020	2656816
16	moshe	29/01/2020	792236
17	yair	17/04/2020	50000
18	noa	05/05/2020	1210428
19	yair	13/05/2020	2699

Variables

Name	Value
start_date	19/06/2018

OK Cancel Clear

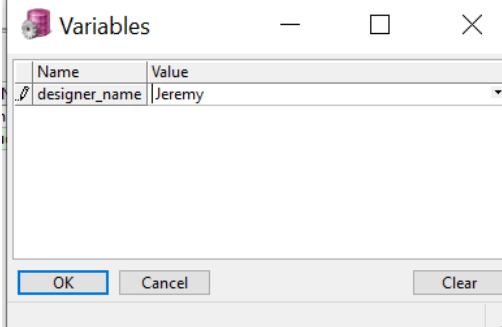
Enter the start date (DD/MM/YYYY)

13:1 debbit@XE Initializing...

בס"ד

הסבר- שאילתה זו מאחזרת את כל ההזמנות ומסננת את התוצאות על סמך מעצב שבבחרה על ידי המשתמש. מתוך רשימת מעצבים

```
SELECT
    c.ClientName,
    i.InvitationId,
    i.Datte,
    i.FinalPrice,
    d.DesignerName,
    p.PackageName
FROM
    Client c
JOIN
    Invitation i ON c.ClientId = i.ClientId
JOIN
    Designer d ON i.DesignerId = d.DesignerID
JOIN
    Pakcage p ON i.PackageId = p.PackageId
WHERE
    d.DesignerID = &<name="designer_name"
                list="SELECT DesignerID, DesignerName FROM Designer ORDER BY DesignerName"
                description="yes" restricted="yes">
ORDER BY
    i.Datte DESC,
    i.FinalPrice DESC;
```



Variables

Name	Value
designer_name	Jeremy

OK Cancel Clear

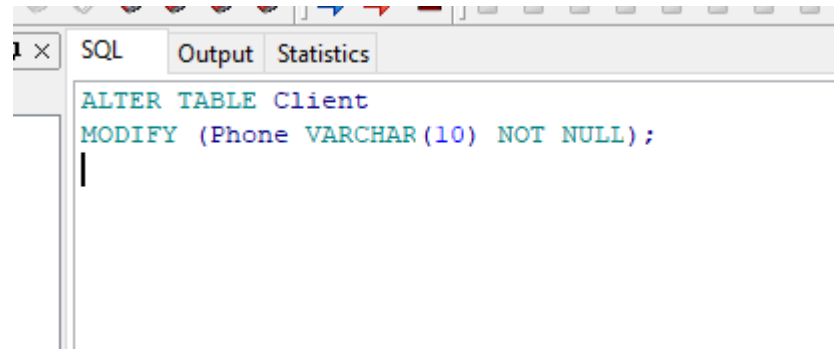
תוצאה

	CLIENTNAME	INVITATIONID	DATTE	FINALPRICE	DESIGNERNAME	PACKAGENAME
1	dani	154	12/02/2021	5524474	Jeremy	Bardesign
2	mali	1	19/06/2018	925483	Jeremy	Bridalbouquet

בס"ד

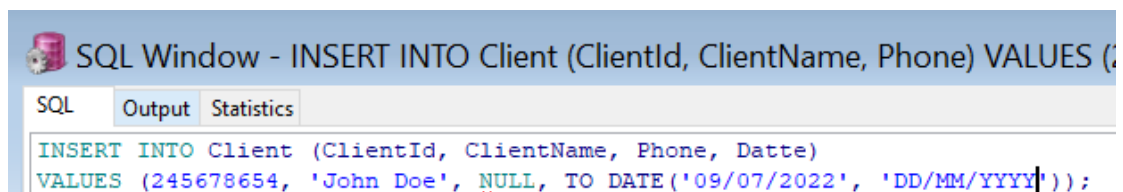
## אילוצים

הסבר: אילוץ זה מבטיח שלעמודת הטלפון בטבלת הלקוח לא יהיו ערכי NULL. לכל לקוח יש לציין מספר טלפון.

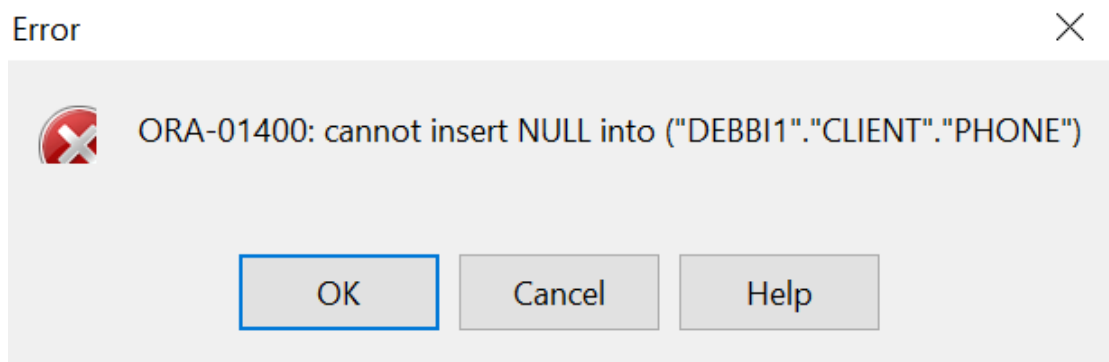


```
SQL
ALTER TABLE Client
MODIFY (Phone VARCHAR(10) NOT NULL);
```

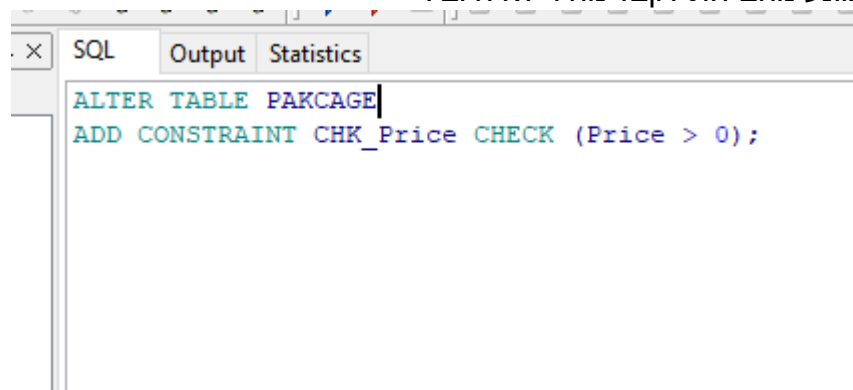
ניסיון:



```
SQL Window - INSERT INTO Client (ClientId, ClientName, Phone) VALUES (
SQL
INSERT INTO Client (ClientId, ClientName, Phone, Date)
VALUES (245678654, 'John Doe', NULL, TO_DATE('09/07/2022', 'DD/MM/YYYY'));
```

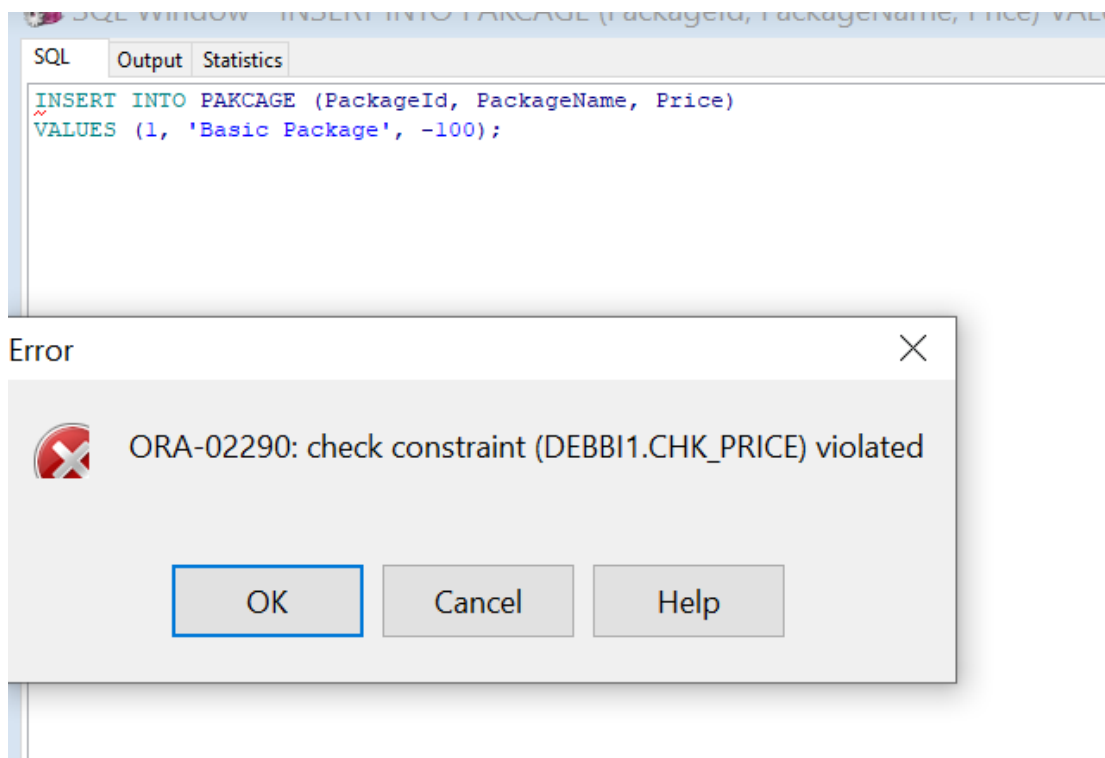


הסבר: אילוץ זה מבטיח שעמודת המחיר בטבלת החבילה חייבת להיות גדולה מאפס. זה מונע מחבילות לקבל מחיר לא חיובי.

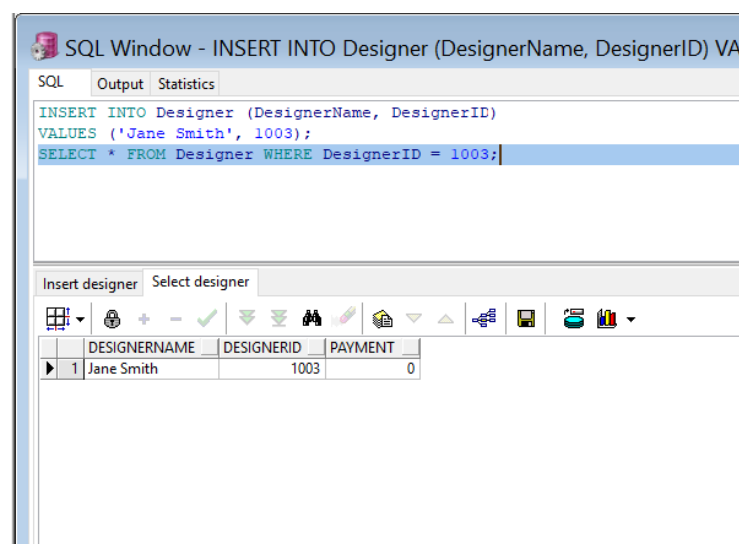
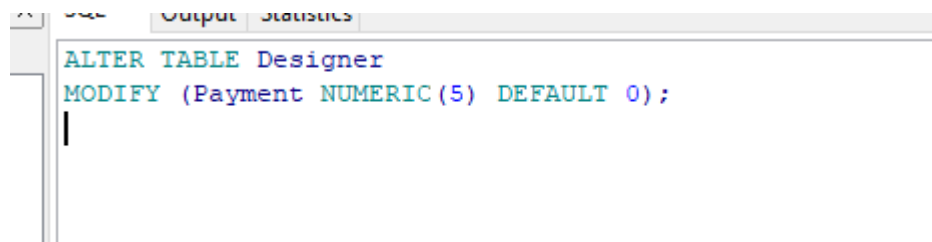


```
SQL
ALTER TABLE PACKAGE
ADD CONSTRAINT CHK_Price CHECK (Price > 0);
```

ניסיון:



הסבר: אילוץ זה מגדיר ערך ברירת מחדל של 0 עבור העמודה תשלום בטבלת מעצב. אם לא צוין סכום תשלום עבור מעצב, ברירת המחדל היא 0.



ניסיון:

## גיבוי

Name	Type	Compiled
CLIENT	TABLE	27/05/2024 13:03:40
CONTAINING	TABLE	26/05/2024 15:19:34
DESIGNER	TABLE	07/07/2024 03:52:39
EQUIPMENT	TABLE	27/05/2024 15:13:53
INSTOCK	TABLE	26/05/2024 15:19:33
INVITATION	TABLE	26/05/2024 15:19:34
PACKAGE	TABLE	07/07/2024 03:23:38
SUPPLIER_	TABLE	26/05/2024 15:19:33

Oracle Export	SQL Inserts	PL/SQL Developer	Log
<input type="checkbox"/> Drop tables	<input checked="" type="checkbox"/> Disable foreign key constraint:		
<input checked="" type="checkbox"/> Create tables	<input checked="" type="checkbox"/> Include storage		
<input type="checkbox"/> Truncate tables	<input checked="" type="checkbox"/> Include privileges		
<input checked="" type="checkbox"/> Delete records	Commit every <input type="text" value="100"/> records (0 = never)		
<input checked="" type="checkbox"/> Disable triggers	Where clause <input type="text"/>		
<input type="checkbox"/> Zip			
Output file <input type="text" value="C:\Users\IMOEO01\minipDataBases\2שלב\backup2.sql"/>			

▼ debbi1@XE Exporting... Done

```
SQL Output Statistics
prompt PL/SQL Developer import file
prompt Created on 2024 יולי 07 יום ראשון by IMOE001
set feedback off
set define off
prompt Creating CLIENT...
create table CLIENT
(
  clientid   VARCHAR2(9) not null,
  clientname VARCHAR2(15) not null,
  phone      VARCHAR2(10) not null
)
tablespace SYSTEM
pctfree 10
pctused 40
initrans 1
maxtrans 255
storage
(
  initial 64K
  next 1M
  minextents 1
  maxextents unlimited
);
alter table CLIENT
add primary key (CLIENTID)
using index
tablespace SYSTEM
pctfree 10
initrans 2
maxtrans 255
storage
(
  initial 64K
  next 1M
  minextents 1
  maxextents unlimited
);

prompt Creating EQUIPMENT...
create table EQUIPMENT
(
  equipmentid   NUMBER(5) not null,
  equipmentname VARCHAR2(50) not null
)
tablespace SYSTEM
pctfree 10
pctused 40
```



## שלב 3

### תוכנית 1

#### פרוצדורה-

הפרוצדורה הנ"ל מוסיפה הזמנה חדשה לטבלה

```
1 CREATE OR REPLACE PROCEDURE add_invitation(  
2     p_client_id IN VARCHAR2,  
3     p_designer_id IN NUMBER,  
4     p_package_id IN NUMBER  
5 ) AS  
6     v_invitation_id NUMBER;  
7     v_final_price NUMBER;  
8 BEGIN  
9     -- Generate a new invitation ID  
10    SELECT NVL(MAX(InvitationId), 0) + 1 INTO v_invitation_id FROM Invitation;  
11  
12    -- Calculate final price (package price + designer payment)  
13    SELECT Price + Payment INTO v_final_price  
14    FROM Package p  
15    JOIN Designer d ON d.DesignerID = p_designer_id  
16    WHERE p.PackageId = p_package_id;  
17  
18    -- Insert new invitation  
19    INSERT INTO Invitation (InvitationId, Date, ClientId, DesignerId, PackageId, FinalPrice)  
20    VALUES (v_invitation_id, SYSDATE, p_client_id, p_designer_id, p_package_id, v_final_price);  
21  
22    -- Print basic invitation details  
23    DBMS_OUTPUT.PUT_LINE('New Invitation Created: ID ' || v_invitation_id || ', Price: $' || v_final_price);  
24  
25 EXCEPTION  
26 WHEN OTHERS THEN  
27     DBMS_OUTPUT.PUT_LINE('Error adding invitation: ' || SQLERRM);  
28     ROLLBACK;  
29 END;  
30  
31
```

TEST

The screenshot shows a database test script editor with a toolbar at the top containing icons for running, saving, and other functions. Below the toolbar, there are tabs for 'Test script', 'DBMS Output', 'Statistics', 'Profiler', and 'Trace'. The 'Test script' tab is active, displaying a PL/SQL script:

```
1 begin
2   -- Call the procedure
3   add_invitation(p_client_id => :p_client_id,
4                 p_designer_id => :p_designer_id,
5                 p_package_id => :p_package_id);
6 end;
```

Below the script editor, there is a table with 4 columns: Variable, Type, and Value. The table contains three rows of variables:

Variable	Type	Value
p_client_id	String	22
p_designer_id	Float	2
p_package_id	Float	4

תוצאה

The screenshot shows the 'DBMS Output' window. It has a toolbar with a 'Clear' button, a 'Buffer size' dropdown set to '10000', and a checkbox labeled 'Enabled' which is checked. Below the toolbar, the output text reads:

```
New Invitation Created: ID 233, Price: $15000
```

## פונקציה

פונקציה לחישוב הערך הכולל של ציוד עבור ספק:

Parameter list	
1	CREATE OR REPLACE FUNCTION calculateS(p_supplier_id IN NUMBER)
2	RETURN NUMBER
3	IS
4	v_total_value NUMBER := 0;
5	
6	CURSOR c_inventory IS
7	SELECT i.Amount, p.Price
8	FROM InStock i
9	JOIN containing c ON i.EquipmentId = c.EquipmentId
10	JOIN Package p ON c.PackageId = p.PackageId
11	WHERE i.SupplierId = p_supplier_id;
12	
13	TYPE t_inventory_item IS RECORD (
14	amount NUMBER,
15	price NUMBER
16	);
17	v_inventory_item t_inventory_item;
18	
19	BEGIN
20	OPEN c_inventory;
21	LOOP
22	FETCH c_inventory INTO v_inventory_item;
23	EXIT WHEN c_inventory%NOTFOUND;
24	
25	v_total_value := v_total_value + (v_inventory_item.amount * v_inventory_item.price);
26	END LOOP;
27	CLOSE c_inventory;
28	
29	RETURN v_total_value;
30	EXCEPTION
31	WHEN OTHERS THEN
32	DBMS_OUTPUT.PUT_LINE('Error: '    SQLERRM);
33	RETURN NULL;
34	END;
35	

## תוצאה

	Variable	Type	Value
<input checked="" type="checkbox"/>	result	Float	2100273500
<input checked="" type="checkbox"/>	p_supplier_id	Float	12345
<input checked="" type="checkbox"/>	*		

## פונקציה נוספת לחישוב ההזמנה הגבוהה ביותר (מקבלת 2 מערכים של הזמנות ומחירים)

```

1 CREATE OR REPLACE FUNCTION get_price(
2   p_invitation_ids IN VARCHAR2,
3   p_prices IN VARCHAR2
4 ) RETURN VARCHAR2
5 IS
6   TYPE t_numbers IS TABLE OF NUMBER INDEX BY PLS_INTEGER;
7   v_ids t_numbers;
8   v_prices t_numbers;
9   v_max_price NUMBER := 0;
10  v_max_id NUMBER := 0;
11  v_index PLS_INTEGER := 1;
12 BEGIN
13   -- Parse the input strings into arrays
14   FOR i IN (SELECT REGEXP_SUBSTR(p_invitation_ids, '[^,]+', 1, LEVEL) AS val
15             FROM DUAL
16             CONNECT BY REGEXP_SUBSTR(p_invitation_ids, '[^,]+', 1, LEVEL) IS NOT NULL)
17   LOOP
18     v_ids(v_index) := TO_NUMBER(i.val);
19     v_index := v_index + 1;
20   END LOOP;
21
22   v_index := 1;
23   FOR i IN (SELECT REGEXP_SUBSTR(p_prices, '[^,]+', 1, LEVEL) AS val
24             FROM DUAL
25             CONNECT BY REGEXP_SUBSTR(p_prices, '[^,]+', 1, LEVEL) IS NOT NULL)
26   LOOP
27     v_prices(v_index) := TO_NUMBER(i.val);
28     v_index := v_index + 1;
29   END LOOP;
30
31   -- Find the highest price and corresponding ID
32   FOR i IN 1..v_prices.COUNT LOOP
33     IF v_prices(i) > v_max_price THEN
34       v_max_price := v_prices(i);
35       v_max_id := v_ids(i);
36     END IF;
37   END LOOP;
38
39   RETURN 'Highest Price: $' || v_max_price || ', Invitation ID: ' || v_max_id;
40 EXCEPTION
41   WHEN OTHERS THEN
42     RETURN 'Error finding highest priced order: ' || SQLERRM;
43 END;

```

debbil@XE

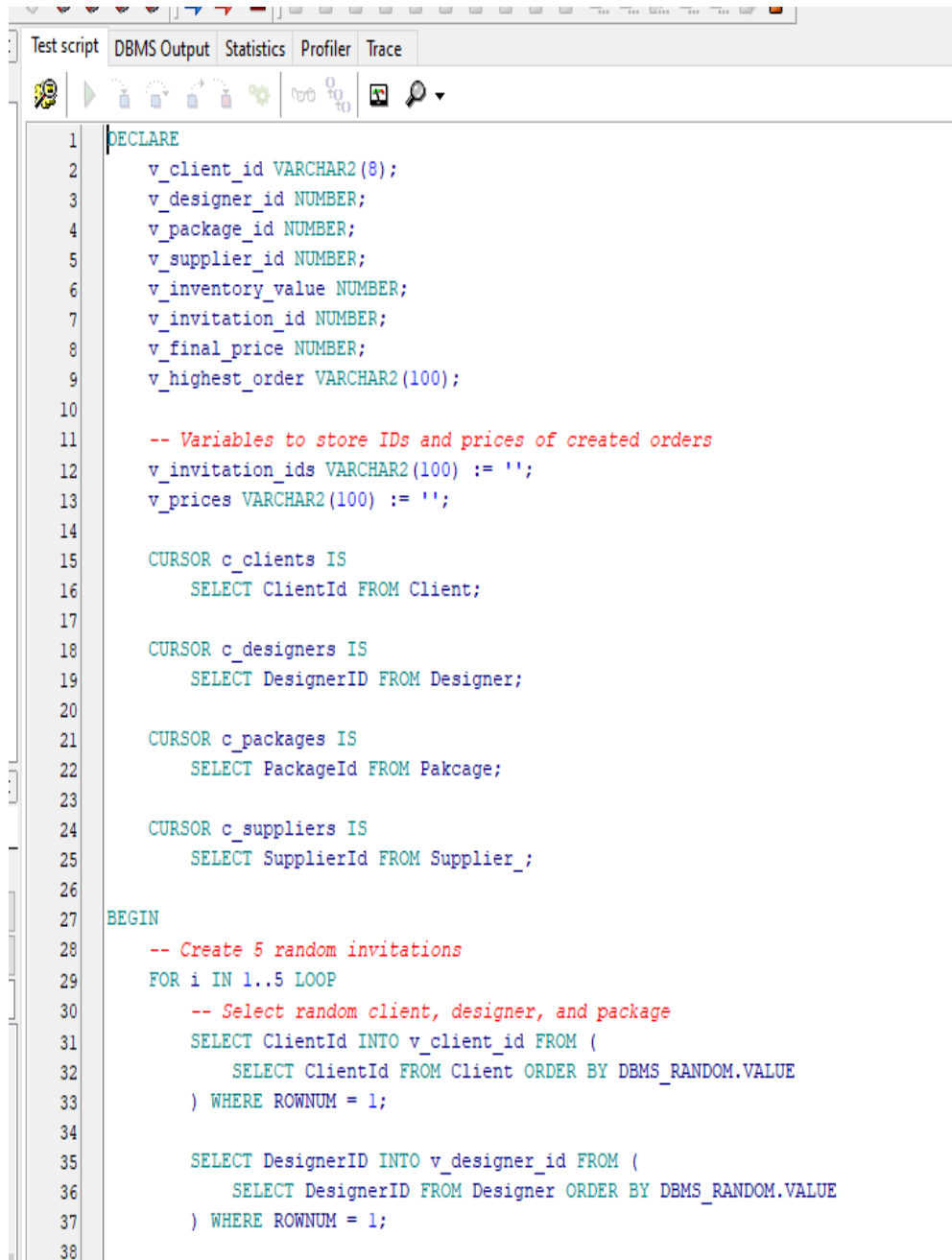
	Variable	Type	Value
<input checked="" type="checkbox"/>	result	String	Highest Price: \$100000, Invitation ID: 1
<input checked="" type="checkbox"/>	p_invitation_ids	String	1,2,3,4
<input checked="" type="checkbox"/>	p_prices	String	100000,23456,76584,87659
<input checked="" type="checkbox"/>			

## תוכנית ראשית

התוכנית יוצרת 5 הזמנות אקראיות תוך שימוש בפרוצדורה שכתבנו בנוסף התוכנית מחשבת ומדפיסה ערך מלאי בעבור כל ספק ומדפיסה מה ערך ציוד הכולל שלו . שידע האם לאחר שלקוחות הזמינו מהמערכת האם להוסיף האם לשפר וכו.

לאחר יצירת ההזמנות, הוא קורא לפונקציה 'get\_price'.

- ומדפיס את ההזמנה הגדולה ביותר במחיר הגבוה ביותר



```
1 DECLARE
2     v_client_id VARCHAR2(8);
3     v_designer_id NUMBER;
4     v_package_id NUMBER;
5     v_supplier_id NUMBER;
6     v_inventory_value NUMBER;
7     v_invitation_id NUMBER;
8     v_final_price NUMBER;
9     v_highest_order VARCHAR2(100);
10
11     -- Variables to store IDs and prices of created orders
12     v_invitation_ids VARCHAR2(100) := '';
13     v_prices VARCHAR2(100) := '';
14
15     CURSOR c_clients IS
16         SELECT ClientId FROM Client;
17
18     CURSOR c_designers IS
19         SELECT DesignerID FROM Designer;
20
21     CURSOR c_packages IS
22         SELECT PackageId FROM Package;
23
24     CURSOR c_suppliers IS
25         SELECT SupplierId FROM Supplier;
26
27 BEGIN
28     -- Create 5 random invitations
29     FOR i IN 1..5 LOOP
30         -- Select random client, designer, and package
31         SELECT ClientId INTO v_client_id FROM (
32             SELECT ClientId FROM Client ORDER BY DBMS_RANDOM.VALUE
33         ) WHERE ROWNUM = 1;
34
35         SELECT DesignerID INTO v_designer_id FROM (
36             SELECT DesignerID FROM Designer ORDER BY DBMS_RANDOM.VALUE
37         ) WHERE ROWNUM = 1;
```

```

SELECT PackageId INTO v_package_id FROM (
    SELECT PackageId FROM Package ORDER BY DBMS_RANDOM.VALUE
) WHERE ROWNUM = 1;

-- Call the add invitation procedure
add_invitation(v_client_id, v_designer_id, v_package_id);
-- Get the invitation ID and final price of the just-created invitation
SELECT InvitationId, FinalPrice
INTO v_invitation_id, v_final_price
FROM Invitation
WHERE InvitationId = (SELECT MAX(InvitationId) FROM Invitation);

-- Store the ID and price
v_invitation_ids := v_invitation_ids || v_invitation_id || ',';
v_prices := v_prices || v_final_price || ',';

DBMS_OUTPUT.PUT_LINE('Created Invitation ID: ' || v_invitation_id || ', Price: $' || v_final_price);
DBMS_OUTPUT.PUT_LINE('-----');
END LOOP;

-- Remove trailing commas
v_invitation_ids := RTRIM(v_invitation_ids, ',');
v_prices := RTRIM(v_prices, ',');

-- Calculate and print inventory value for each supplier
FOR supplier IN c_suppliers LOOP
    v_inventory_value := calculateS(supplier.SupplierId);
    DBMS_OUTPUT.PUT_LINE('Supplier ID ' || supplier.SupplierId || ' inventory value: $' || v_inventory_value);
END LOOP;

-- Find and print the highest priced order among the 5 created orders
v_highest_order := get_price(v_invitation_ids, v_prices);
DBMS_OUTPUT.PUT_LINE('-----');
DBMS_OUTPUT.PUT_LINE('Highest Priced Order: ' || v_highest_order);

EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
END;
```

```

Test script DBMS Output Statistics Profiler Trace
Clear Buffer size 10000 ✓ Enabled

New Invitation Created: ID 249, Price: $11700
Created Invitation ID: 249, Price: $11700
-----
New Invitation Created: ID 250, Price: $16500
Created Invitation ID: 250, Price: $16500
-----
New Invitation Created: ID 251, Price: $15300
Created Invitation ID: 251, Price: $15300
-----
New Invitation Created: ID 252, Price: $8000
Created Invitation ID: 252, Price: $8000
-----
New Invitation Created: ID 253, Price: $14400
Created Invitation ID: 253, Price: $14400
-----
Supplier ID 12345 inventory value: $2100273500
Supplier ID 12365 inventory value: $3153223500
Supplier ID 12395 inventory value: $565155000
Supplier ID 14565 inventory value: $2168563000
Supplier ID 15365 inventory value: $1605259500
Supplier ID 19475 inventory value: $1605005500
Supplier ID 23565 inventory value: $1463842000
Supplier ID 87655 inventory value: $1603344000
Supplier ID 89065 inventory value: $1214574000
Supplier ID 89365 inventory value: $1487025000
-----
Highest Priced Order: Highest Price: $16500, Invitation ID: 250

```

## תוכנית 2

### פרוצדורה

מעדכן את תשלום המעצב על סמך הביצועים שלו  
(מקבל מזהה מעצב ואחוז להעלאת השכר)

```
1 CREATE OR REPLACE PROCEDURE update_designer_payment(  
2     p_designer_id IN NUMBER,  
3     p_increase_percentage IN NUMBER  
4 ) AS  
5     v_current_payment NUMBER;  
6     v_new_payment NUMBER;  
7 BEGIN  
8     -- Get current payment  
9     SELECT Payment INTO v_current_payment  
10    FROM Designer  
11   WHERE DesignerID = p_designer_id;  
12  
13     -- Calculate new payment  
14     v_new_payment := v_current_payment * (1 + p_increase_percentage / 100);  
15  
16     -- Update designer payment  
17     UPDATE Designer  
18     SET Payment = v_new_payment  
19     WHERE DesignerID = p_designer_id;  
20  
21     DBMS_OUTPUT.PUT_LINE('Designer ' || p_designer_id || ' payment updated from $' ||  
22                          v_current_payment || ' to $' || v_new_payment);  
23 EXCEPTION  
24     WHEN NO_DATA_FOUND THEN  
25         DBMS_OUTPUT.PUT_LINE('Designer ' || p_designer_id || ' not found');  
26     WHEN OTHERS THEN  
27         DBMS_OUTPUT.PUT_LINE('Error updating designer payment: ' || SQLERRM);  
28 END;  
29
```

### תוצאה

Test script DBMS Output Statistics Profiler Trace

Clear Buffer size 10000 ☒ Enabled

Designer 2 payment updated from \$8000 to \$10400

	Variable	Type	Value
<input checked="" type="checkbox"/>	p_designer_id	Float	2
<input checked="" type="checkbox"/>	p_increase_percentage	Float	30
<input checked="" type="checkbox"/>	*		



## פונקציה

פונקציה זו מחשבת את מידת הפופולריות של חבילה על סמך כמה פעמים היא הוזמנה.


מה שזה עושה:

1. זה סופר את מספר הפעמים שהחבילה שצוינה מופיעה בטבלת ההזמנות.

2. הוא מחזיר את הספירה הזו בתור ציון הפופולריות.

```
1 CREATE OR REPLACE FUNCTION get_package_popularity(  
2     p_package_id IN NUMBER  
3 ) RETURN NUMBER AS  
4     v_popularity NUMBER;  
5 BEGIN  
6     SELECT COUNT(*) INTO v_popularity  
7     FROM Invitation  
8     WHERE PackageId = p_package_id;  
9  
10    RETURN v_popularity;  
11 EXCEPTION  
12 WHEN NO_DATA_FOUND THEN  
13     RETURN 0;  
14 WHEN OTHERS THEN  
15     DBMS_OUTPUT.PUT_LINE('Error calculating package popularity: ' || SQLERRM);  
16     RETURN -1;  
17 END;  
18
```

## תוצאה

<input type="checkbox"/>	Variable	Type	Value
<input checked="" type="checkbox"/>	result	Float	20
<input checked="" type="checkbox"/>	p_package_id	Float	3
<input checked="" type="checkbox"/>	*		
	1:1	0:01	debbi1@XE Executed in 1.953 seconds

## תוכנית ראשית

התוכנית הראשית מנתחת את ביצועי המעצבים, מעדכנת את התשלומים שלהם ובודקת את הפופולריות של החבילות.

מה שזה עושה: ביצועי מעצבים ועדכון תשלום:

- זה עובר לולאה דרך כל המעצבים באמצעות סמן.

- לכל מעצב:

א. זה סופר כמה הזמנות יצר המעצב.

ב. בהתבסס על ספירה זו, הוא קובע גידול באחוזים עבור התשלום שלהם:

- עלייה של 10% אם הם יצרו יותר מ-10 הזמנות

- עלייה של 5% אם הם יצרו יותר מ-5 אך לא יותר מ-10 הזמנות

- אין עלייה (0%) אם הם יצרו 5 הזמנות או פחות

ג. זה קורא לנוהל update\_designer\_payment עם מזהה המעצב ואחוז העלייה המחושב.

לאחר מכן מנתח פופולריות של כל חבילה.

```
DECLARE
    v_designer_id NUMBER;
    v_package_id NUMBER;
    v_designer_invitations NUMBER;
    v_package_popularity NUMBER;
    v_increase_percentage NUMBER;

    CURSOR c_designers IS
        SELECT DesignerID
        FROM Designer;

    CURSOR c_packages IS
        SELECT PackageId
        FROM Package;

BEGIN
    -- Analyze designer performance and update payments
    FOR designer IN c_designers LOOP
        -- Count invitations for this designer
        SELECT COUNT(*) INTO v_designer_invitations
        FROM Invitation
        WHERE DesignerId = designer.DesignerID;

        -- Calculate payment increase based on performance
        v_increase_percentage := CASE
            WHEN v_designer_invitations > 10 THEN 10
            WHEN v_designer_invitations > 5 THEN 5
            ELSE 0
        END;

        -- Update designer payment
        update_designer_payment(designer.DesignerID, v_increase_percentage);
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('-----');
```

```

-- Analyze package popularity
FOR package IN c_packages LOOP
    v_package_popularity := get_package_popularity(package.PackageId);

    DBMS_OUTPUT.PUT_LINE('Package ' || package.PackageId ||
                          ' popularity: ' || v_package_popularity || ' orders');
END LOOP;

EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error in main program: ' || SQLERRM);
END;

```

test script DBMS Output Statistics Profiler Trace

Clear Buffer size 10000 ☒ Enabled

```

Designer 172 payment updated from $8600 to $8600
Designer 173 payment updated from $5700 to $5700
Designer 174 payment updated from $9400 to $9400
Designer 175 payment updated from $5900 to $5900
Designer 176 payment updated from $8300 to $8300
Designer 177 payment updated from $6100 to $6100
Designer 178 payment updated from $8700 to $8700
Designer 179 payment updated from $6200 to $6200
Designer 180 payment updated from $8800 to $8800
Designer 181 payment updated from $6000 to $6000
Designer 182 payment updated from $9000 to $9000
Designer 183 payment updated from $5900 to $5900
Designer 184 payment updated from $8600 to $8600
Designer 185 payment updated from $6100 to $6100
Designer 186 payment updated from $9100 to $9100
Designer 187 payment updated from $5700 to $5700
Designer 188 payment updated from $8300 to $8300
Designer 189 payment updated from $6000 to $6000
Designer 190 payment updated from $9400 to $9400
Designer 191 payment updated from $5800 to $5800
Designer 192 payment updated from $8200 to $8200
Designer 193 payment updated from $6100 to $6100
Designer 194 payment updated from $8700 to $8700
Designer 195 payment updated from $6000 to $6000
Designer 196 payment updated from $9100 to $9100
Designer 197 payment updated from $5900 to $5900
Designer 198 payment updated from $8600 to $8600
Designer 199 payment updated from $6200 to $6200
Designer 200 payment updated from $9400 to $9400
Designer 1001 payment updated from $0 to $0
Designer 1002 payment updated from $0 to $0
Designer 1003 payment updated from $0 to $0
-----
Package 1 popularity: 25 orders
Package 2 popularity: 25 orders
Package 3 popularity: 20 orders
Package 4 popularity: 25 orders
Package 6 popularity: 20 orders
Package 7 popularity: 29 orders
Package 8 popularity: 18 orders
Package 9 popularity: 19 orders
Package 10 popularity: 19 orders
Package 11 popularity: 0 orders

```

**תוצאה**

Name	Type	Compiled
CLIENT	TABLE	27/05/2024 13:03:40
CONTAINING	TABLE	26/05/2024 15:19:34
DESIGNER	TABLE	07/07/2024 03:52:39
EQUIPMENT	TABLE	27/05/2024 15:13:53
INSTOCK	TABLE	26/05/2024 15:19:33
INVITATION	TABLE	26/05/2024 15:19:34
PACKAGE	TABLE	07/07/2024 03:23:38
SUPPLIER_	TABLE	26/05/2024 15:19:33

Oracle Export

SQL Inserts

PL/SQL Developer

Log

☐ Drop tables
 ☒ Create tables
 ☐ Truncate tables
 ☒ Delete records
 ☒ Disable triggers
 ☐ Zip

☒ Disable foreign key constraint:
 ☒ Include storage
 ☒ Include privileges
 Commit every 100 records (0 = never)
 Where clause

Output file C:\Users\IMOE001\minipDataBases\שלב 3\backup3.sql

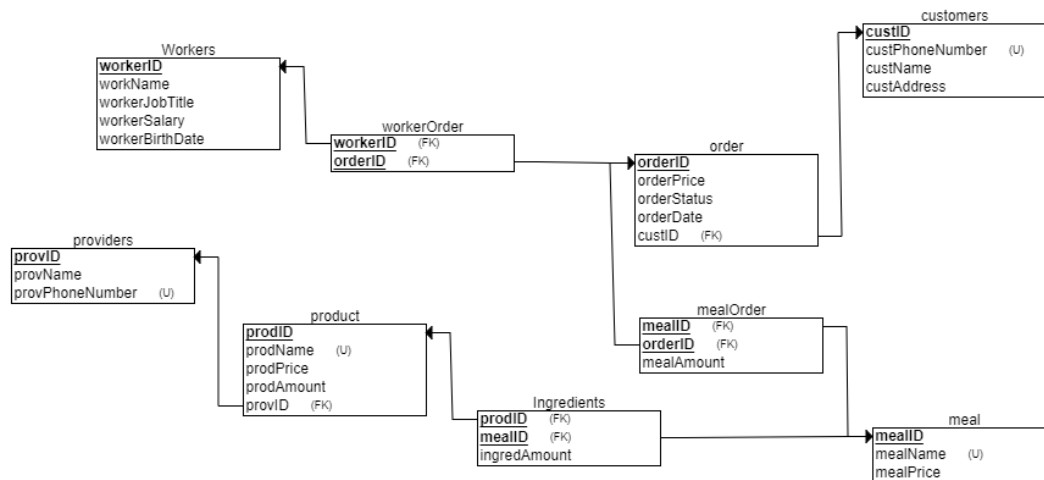
debbi1@XE
 

Exporting... Done

## שלב 4

קיבלנו את פרויקט " מסעדה"

### DSD



### מסקנות מDSD כדי ליצור את הERD-

#### טבלת CUSTOMERS

3 עמודות מקושר לטבלת ORDER, מפתח ראשי-CUSTID

#### טבלת MEAL

3 עמודות מקושר לטבלת PRODUCT, מפתח ראשי-MEALID

#### טבלת INGREDIENTS- טבלת קשר

3 עמודות

מפתח ראשי – 2 מפתחות זרים MEALID, PRODID מחובר לטבלאות MEAL, PRODUCT

#### טבלת PRODUCT

5 עמודות מפתח ראשי PRODID מקושר לPROVIDERS בקשר של רבים ליחיד ולטבלת MEAL בקשר של רבים לרבים

#### טבלת PROVIDERS

3 עמודות מפתח ראשי PROVIDID מקושר לטבלת PRODUCT

#### טבלת MEALORDER – טבלת קשר

מחוברת לטבלאות MEAL, ORDER

3 עמודות מפתח ראשי – 2 מפתחות זרים MEALID, ORDERID

## טבלת ORDERS

5 עמודות מפתח ראשי ORDERID

מחוברת לטבלת CUSTOMERS, לטבלת MEAL, ולטבלת WORKERS

## טבלת WORKERS

5 עמודות מפתח ראשי WORKERID מחובר לטבלת ORDER

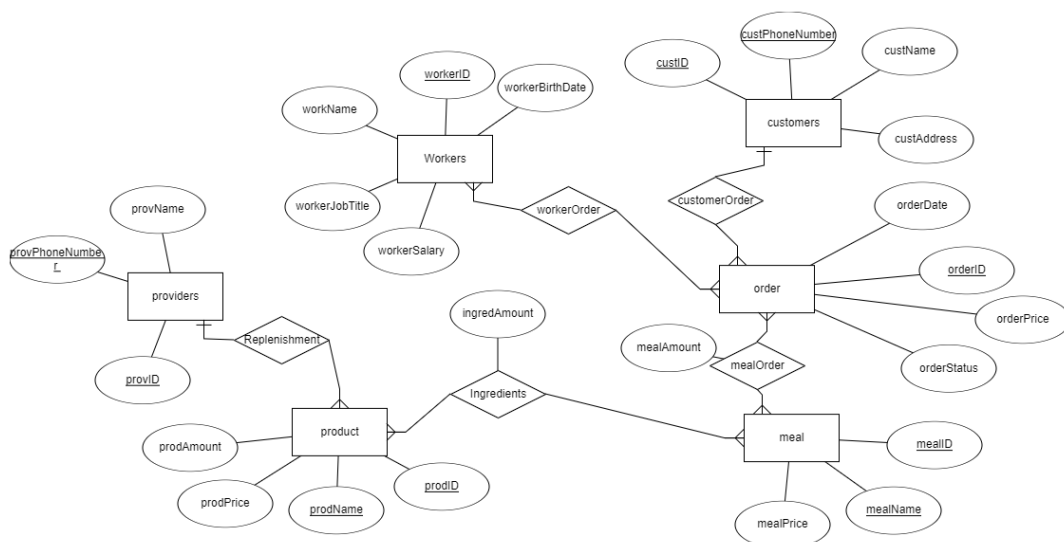
## טבלת WORKERORDER

טבלת קשר 2 עמודות, מפתח ראשי – 2 מפתחות זרים ORDERID, WORKERID

מחובר לטבלאות WORKERS וORDER

שמות הישיות הן שמות הטבלאות. שמות התכונות הן שמות העמודות בהתאם לכל טבלה.  
מצוין 2 טבלאות שמציינות קשר של רבים ליחיד, כל שאר הקשרים מציינים קשר של רבים לרבים.

## ERD



**פקודת עיצוב 1**

מאחר ובשני הפרוייקטים יש טבלה המייצגת לקוחות, מחקנו את הטבלה של הלקוחות מהפרוייקט שקיבלנו והעברנו את כל הקשרים שהיו מחוברים לטבלה זו, אל הטבלה של הלקוחות שלנו, בהתאם לשלבים המוצגים בקוד הבא העברנו את הקשר של טבלת ORDERS מטבלת CUSTOMERS של הפרוייקט שקיבלנו אל טבלת CUSTOMER של הפרוייקט שלנו.

**CLIENT-CUSTOMER**

```

SQL Output Statistics
-- Modify Client table
ALTER TABLE Client
ADD Address VARCHAR(30);
-- Increase the size of the PhoneNumber column to accommodate larger values
ALTER TABLE Client
MODIFY Phone VARCHAR(20);
ALTER TABLE Client
MODIFY ClientName VARCHAR(40);
-- הכנסו הערכים מהטבלה שלנו לטבלה שלנו
INSERT INTO Client (ClientId, ClientName, Phone, BirthDate, Address)
SELECT custID, custName, custPhoneNumber, to_date('01/01/1990','dd/mm/yyyy'), custAddress FROM Customers
WHERE custID NOT IN (SELECT ClientId FROM Client);
-- Update custAddress to "Dodaim 5 Ramat Gan" where it is NULL
UPDATE Client
SET Address = 'Dodaim 5 Ramat Gan'
WHERE Address IS NULL;
-- Update BirthDate to "01/01/2007" where it is NULL
UPDATE Client
SET BirthDate = TO_DATE('01/01/2007', 'DD/MM/YYYY')
WHERE BirthDate IS NULL;
-- שינוי תאריך כד נשכח להעביר מפתח זר
ALTER TABLE Orders
ADD tempCustID VARCHAR2(10);
UPDATE Orders
SET tempCustID = TO_CHAR(custID);
ALTER TABLE Orders
DROP COLUMN custID;
ALTER TABLE Orders
RENAME COLUMN tempCustID TO custID;
-- foreign key
-- 1. Find the correct constraint name
SELECT constraint_name
FROM user_constraints
WHERE table_name = 'ORDERS'
AND constraint_type = 'R';
-- 2. Drop the correct foreign key constraint
ALTER TABLE Orders
DROP CONSTRAINT SYS_C007341;
-- 3. Add the new foreign key constraint
ALTER TABLE Orders
ADD CONSTRAINT fk_orders_client
FOREIGN KEY (custID) REFERENCES Client(ClientId);
DROP TABLE Customers;

select * from Client t

```

Done in 0.156 seconds

## פקודת עיצוב 2

מאחר ובשני הפרוייקטים יש טבלה המייצגת ספקים-**PROVIDES-SUPPLIER**, מחקנו מהפרייקט שלנו את הטבלה המייצגת ספקים מהפרייקט השני והעברנו את כל התכונות הנוספות ועדכנו אל הטבלה של הספקים מהפרייקט שלנו בהתאם לפקודות בקוד הבא התאמנו את הטבלאות

העברנו את כל הקשרים שהיו מחוברים לטבלה זו, אל הטבלה של הלקוחות שלנו, העברנו את הקשר מטבלת PRODUCT אל הטבלה החדשה

```
SQL Output Statistics
-- Add new column provPhoneNumber if it doesn't already exist
ALTER TABLE Supplier_
ADD (provPhoneNumber VARCHAR2(20));

-- Modify existing columns to ensure they meet requirements
ALTER TABLE Supplier_
MODIFY (SupplierId NUMBER(10));
ALTER TABLE Supplier_
MODIFY (SupplierName VARCHAR2(20));

INSERT INTO Supplier_ (SupplierId, SupplierName, Area, provPhoneNumber)
SELECT provID, SUBSTR(provName, 1, 20), NULL, provPhoneNumber
FROM Providers
WHERE provID NOT IN (SELECT SupplierId FROM Supplier_);

-- Update Area with random values where it is NULL
UPDATE Supplier_
SET Area =
CASE
WHEN DBMS_RANDOM.VALUE < 0.33 THEN 'North'
WHEN DBMS_RANDOM.VALUE < 0.66 THEN 'South'
ELSE 'gushDan'
END
WHERE Area IS NULL;

-- Update provPhoneNumber with random values where it is NULL
UPDATE Supplier_
SET provPhoneNumber = '0' || LPAD(TRUNC(DBMS_RANDOM.VALUE(100000000, 999999999)), 9, '0')
WHERE provPhoneNumber IS NULL;

--forigen key
-- 1. Find the correct constraint name
SELECT constraint_name
FROM user_constraints
WHERE table_name = 'PRODUCT'
AND constraint_type = 'R';

-- 2. Drop the correct foreign key constraint
ALTER TABLE Product
DROP CONSTRAINT SYS_C007328;

-- 3. Add the new foreign key constraint
ALTER TABLE Product
ADD CONSTRAINT FK_PRODUCT_SUPPLIER
FOREIGN KEY (provID) REFERENCES Supplier_ (SupplierId);
```

17:1 debbi1@XE SQL script saved successfully



בס"ד

### פקודת עיצוב 3

מאחר ובשני הפרוייקטים יש טבלה המייצגת הזמנות-**invattion-order**, מחקנו מהפרוייקט שלנו את הטבלה המייצגת הזמנות מהפרוייקט השני והעברנו את כל התכונות הנוספות ועדכנו אל הטבלה של ההזמנות מהפרוייקט שלנו בהתאם לפקודות בקוד הבא התאמנו את הטבלאות

מכיוון שבטבלה שלנו יש מעצב ואנו זקוקים לו בשביל ההזמנות לעיצוב ארועים החלטנו שמעצב מספר 1 יהיה זה שאחראי על מסעדות

העברנו את כל הקשרים הנדרשים מטבלת MEALORDER\ WORKERORDER

```
SQL Output Statistics
ALTER TABLE Invitation
ADD orderStatus VARCHAR(20);
ALTER TABLE Invitation
MODIFY ClientId VARCHAR(10); -- Or a larger size if needed

ALTER TABLE Invitation
MODIFY PackageId NUMBER NULL; -- Or use the appropriate data type

INSERT INTO Invitation (InvitationId, FinalPrice, Datte, ClientId, DesignerId, PackageId, orderStatus)
SELECT orderID, orderPrice, orderDate, custID, 1, NULL, orderStatus FROM Orders
WHERE orderID NOT IN (SELECT InvitationId FROM Invitation);

UPDATE Invitation
SET orderStatus =
CASE
    WHEN MOD(ROWNUM, 4) = 0 THEN 'received'
    WHEN MOD(ROWNUM, 4) = 1 THEN 'sent'
    WHEN MOD(ROWNUM, 4) = 2 THEN 'delivered'
    ELSE 'in preparation'
END
WHERE orderStatus IS NULL;

select * from INVITATION t
--forigen key
-- 2. Drop the correct foreign key constraint
ALTER TABLE WorkerOrder DROP CONSTRAINT SYS_C007358;
ALTER TABLE mealOrder DROP CONSTRAINT SYS_C007347;
-- 3. Add the new foreign key constraint
ALTER TABLE WorkerOrder
ADD CONSTRAINT fk_workerorder_invitation
FOREIGN KEY (orderID) REFERENCES Invitation(InvitationId);

ALTER TABLE mealOrder
ADD CONSTRAINT fk_mealorder_invitation
FOREIGN KEY (orderID) REFERENCES Invitation(InvitationId);

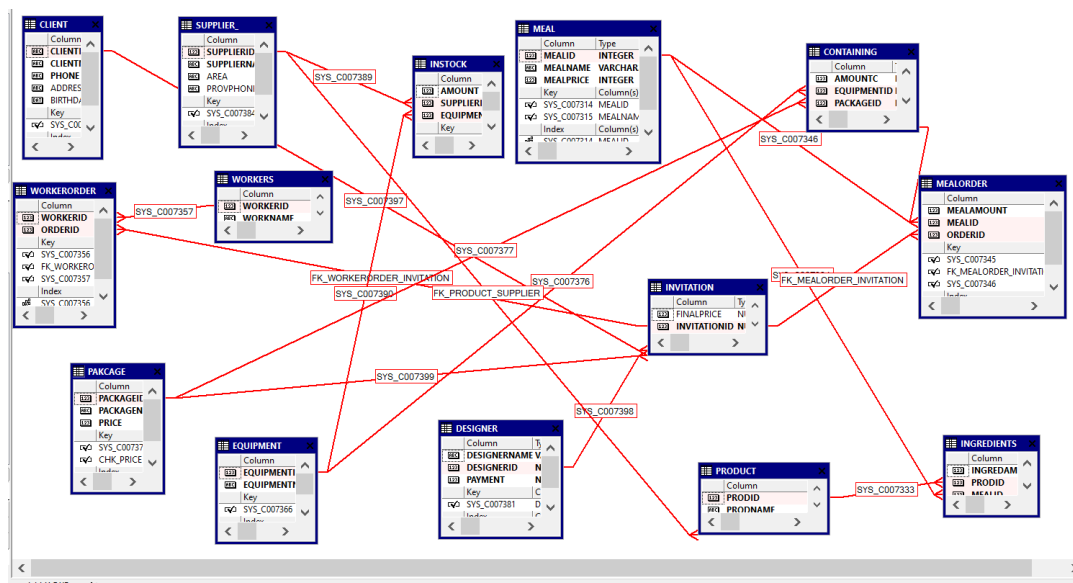
DROP TABLE Orders;
```

דוגמא העברת קשר

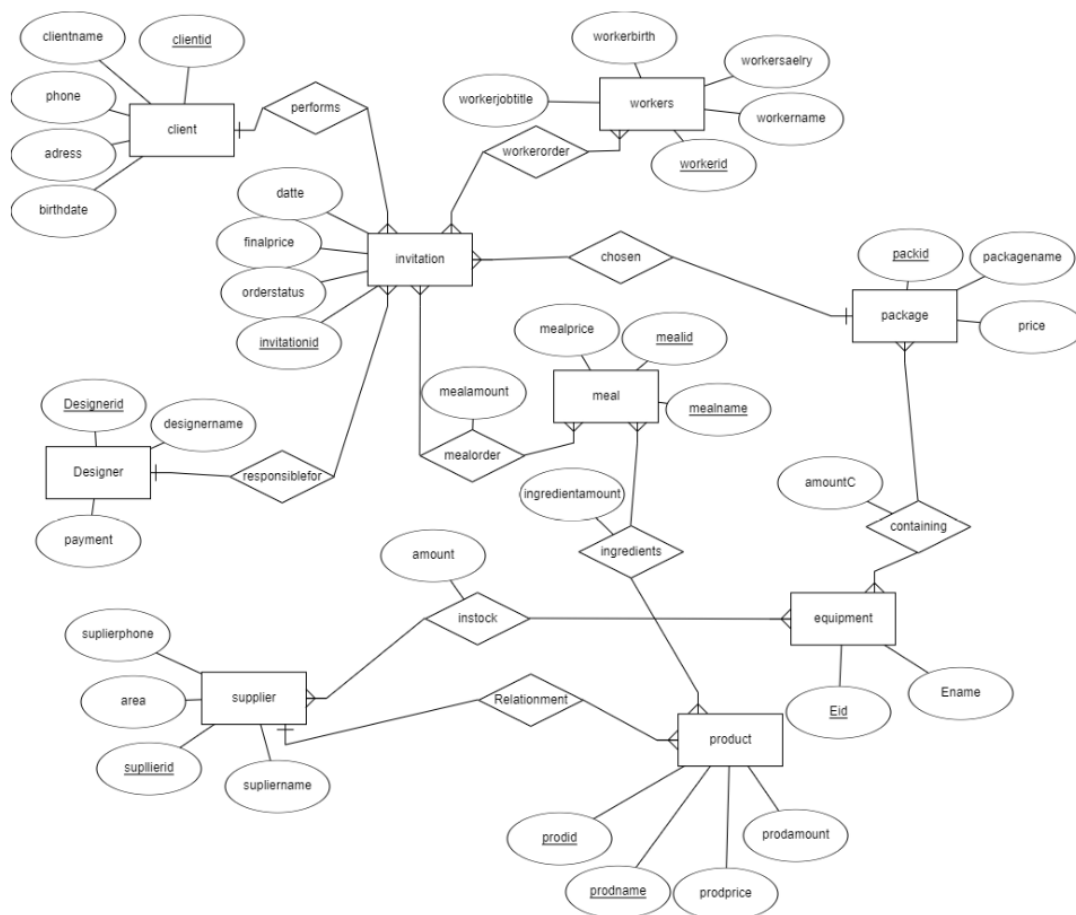
Name	Type	Columns	Enabled	Referencing table	Referencing columns	On Delete	Deferrable	Deferred	Last change
SYS_C007356	Primary	WORKERID, ORDERID	<input checked="" type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>	23/07/2024 11:44:25
FK_WORKERORDER_INVITATION	Foreign	ORDERID	<input checked="" type="checkbox"/>	INVITATION	INVITATIONID	No action	<input type="checkbox"/>	<input type="checkbox"/>	23/07/2024 18:44:29
SYS_C007357	Foreign	WORKERID	<input checked="" type="checkbox"/>	WORKERS	WORKERID	No action	<input type="checkbox"/>	<input type="checkbox"/>	23/07/2024 11:44:25
*			<input checked="" type="checkbox"/>				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

שאר הטבלאות שונות אחת מהשניה ולכן השארנו אותם כמו שהם.

## DSD ממשותף



## ERD משותף



בס"ד

## VIEWS

### מבט מהפרויקט שלנו -עיצוב ארועים

```
CREATE VIEW EventDesignView AS
SELECT
    i.InvitationId,
    i.FinalPrice,
    i.Datte AS EventDate,
    c.ClientId,
    c.ClientName,
    c.Phone AS ClientPhone,
    c.BirthDate AS ClientBirthDate,
    d.DesignerID,
    d.DesignerName,
    d.Payment AS DesignerPayment,
    p.PackageId,
    p.PackageName,
    p.Price AS PackagePrice,
    e.EquipmentId,
    e.EquipmentName,
    co.AmountC AS EquipmentAmountInPackage,
    inStock.Amount AS EquipmentAmountInStock,
    s.SupplierName,
    s.Area AS SupplierArea
FROM Invitation i
JOIN Client c ON i.ClientId = c.ClientId
JOIN Designer d ON i.DesignerID = d.DesignerID
JOIN Package p ON i.PackageId = p.PackageId
JOIN containing co ON p.PackageId = co.PackageId
JOIN Equipment e ON co.EquipmentId = e.EquipmentId
JOIN InStock inStock ON e.EquipmentId = inStock.EquipmentId
JOIN Supplier_ s ON inStock.SupplierId = s.SupplierId;

select * from EventDesignView
```

INVITATIONID	FINALPRICE	EVENTDATE	CLIENTID	CLIENTNAME	CLIENTPHONE	CLIENTBIRTHDATE	DESIGNERID	DESIGNERNAME	DESIGNERPAYMENT	PACKAGEID	PACKAGENAME	PACKAGEPRICE	EQUIPMENTID
1	925483	19/06/2018	34567821	mali	533456778	01/01/2007	164	Jeremy	7800	3	Bridalbouquet	2000	
2	925483	19/06/2018	34567821	mali	533456778	01/01/2007	164	Jeremy	7800	3	Bridalbouquet	2000	
3	925483	19/06/2018	34567821	mali	533456778	01/01/2007	164	Jeremy	7800	3	Bridalbouquet	2000	
4	925483	19/06/2018	34567821	mali	533456778	01/01/2007	164	Jeremy	7800	3	Bridalbouquet	2000	
5	925483	19/06/2018	34567821	mali	533456778	01/01/2007	164	Jeremy	7800	3	Bridalbouquet	2000	
6	925483	19/06/2018	34567821	mali	533456778	01/01/2007	164	Jeremy	7800	3	Bridalbouquet	2000	
7	925483	19/06/2018	34567821	mali	533456778	01/01/2007	164	Jeremy	7800	3	Bridalbouquet	2000	
8	925483	19/06/2018	34567821	mali	533456778	01/01/2007	164	Jeremy	7800	3	Bridalbouquet	2000	

## שאלתה1

### 3 החבילות הפופולריות ביותר עם המחיר הסופי הממוצע וספירת הציווד שלהן

```
SELECT * FROM (
    SELECT
        PackageId,
        PackageName,
        COUNT(DISTINCT InvitationId) AS TimesBooked,
        AVG(FinalPrice) AS AverageFinalPrice,
        COUNT(DISTINCT EquipmentId) AS UniqueEquipmentCount
    FROM
        EventDesignView
    GROUP BY
        PackageId, PackageName
    ORDER BY
        COUNT(DISTINCT InvitationId) DESC
)
WHERE ROWNUM <= 3;
```

PACKAGEID	PACKAGENAME	TIMESBOOKED	AVERAGEFINALPRICE	UNIQUEEQUIPMENTCOUNT
1	7 tablegarlands	29	4249009.55172414	4
2	1 bridachair	25	4967847.76	8
3	2 Tabledecoration	25	3892633.96	7

## שאלתה 2 מחזיר את ההזמנה הזולה ביותר מתאריך מסים שהמשתמש מקליד

The screenshot displays the SQL Server Enterprise Manager interface. The main window shows a SQL query designed to find the cheapest invitation from a specific date using the EventDesignView. The query is as follows:

```
-- Query to find the cheapest invitation from a specific date using EventDesignView
SELECT
    InvitationId,
    FinalPrice,
    EventDate
FROM (
    SELECT
        e.InvitationId,
        e.FinalPrice,
        e.EventDate
    FROM
        EventDesignView e
    WHERE
        e.EventDate = TO_DATE('&startDate', 'dd/mm/yyyy')
    ORDER BY
        e.FinalPrice ASC
)
WHERE ROWNUM = 1;
```

Below the query editor, the results grid shows a single row of data:

	INVITATIONID	FINALPRICE	EVENTDATE
1	81	5085120	27/09/2007

Overlaid on the bottom right is a 'Variables' dialog box. It contains a table with two columns: 'Name' and 'Value'. The first row shows 'startDate' with the value '27/09/2007'. At the bottom of the dialog are 'OK', 'Cancel', and 'Clear' buttons.

## מבט מהפרויקט שקיבלנו:

SQLOutputStatistics

CREATE VIEW RestaurantView AS  
SELECT  
    m.mealID,  
    m.mealName,  
    m.mealPrice,  
    p.prodID,  
    p.prodName,  
    p.prodPrice AS ProductPrice,  
    i.ingredAmount AS ProductAmount,  
    s.SupplierName,  
    s.Area AS SupplierArea  
FROM Meal m  
JOIN Ingredients i ON m.mealID = i.mealID  
JOIN Product p ON i.prodID = p.prodID  
JOIN Supplier\_ s ON p.prodID = s.SupplierId;  
  
select \* from RestaurantView

Create restaurantviewSelect restaurantview

## שאלתה

מוצא את ההזמנה הגבוהה ביותר

SQL Output Statistics

```
-- Query to find the most expensive meal
SELECT
    mealID,
    mealName,
    mealPrice
FROM (
    SELECT
        mealID,
        mealName,
        mealPrice
    FROM
        RestaurantView
    ORDER BY
        mealPrice DESC
)
WHERE ROWNUM = 1;
```

	MEALID	MEALNAME	MEALPRICE
1	364	chicken-skewer	132

בס"ד

## שאלתה 2

שאלתה מחזירה רשימה של כל הארוחות עם עלויות המוצר הכוללות שלהן, מוגבל למספר מסוים 5

The screenshot shows the SQL Developer interface. The top pane displays a SQL query designed to list meals with their total product costs, limited to five results. The query uses a subquery to calculate the total product cost for each meal by joining the RestaurantView, Meal, Product, and Ingredients tables. The bottom pane shows the results of the query, which are displayed in a table with five rows.

```
-- Query to list all meals with their total product costs, limited to a certain number
SELECT *
FROM (
  SELECT
    mealID,
    mealName,
    SUM(ProductPrice * ProductAmount) AS TotalProductCost
  FROM (
    SELECT
      m.mealID,
      m.mealName,
      p.prodPrice AS ProductPrice,
      i.ingredAmount AS ProductAmount
    FROM
      RestaurantView rv
    JOIN Meal m ON rv.mealID = m.mealID
    JOIN Product p ON rv.prodID = p.prodID
    JOIN Ingredients i ON rv.prodID = i.prodID
    WHERE m.mealID = rv.mealID
  )
  GROUP BY
    mealID,
    mealName
  ORDER BY
    TotalProductCost DESC
)
WHERE ROWNUM <= 5; -- Adjust the number as needed
```

	MEALID	MEALNAME	TOTALPRODUCTCOST
1	173	Dumplings	536546
2	774	steak	524168
3	643	beer	518597
4	519	kebab-skewer	461078
5	446	Meatballs	456956

גיבוי

The screenshot shows the 'Export Tables' dialog box in SQL Developer. The dialog lists various tables in the database, including CLIENT, CONTAINING, DESIGNER, EQUIPMENT, INGREDIENTS, INSTOCK, INVITATION, MEAL, MEALORDER, PAKAGE, PRODUCT, SUPPLIER, WORKERORDER, and WORKERS. The 'Export Tables' tab is selected, and the 'Output file' is set to 'C:\Users\IMOE001\minipDataBases\4\שלב 4\backup4.sql'. The 'Export' button is highlighted.

Name	Type	Compiled
CLIENT	TABLE	23/07/2024 18:01:34
CONTAINING	TABLE	23/07/2024 16:28:01
DESIGNER	TABLE	23/07/2024 16:28:01
EQUIPMENT	TABLE	23/07/2024 16:28:01
INGREDIENTS	TABLE	23/07/2024 11:44:29
INSTOCK	TABLE	23/07/2024 16:28:01
INVITATION	TABLE	23/07/2024 18:32:53
MEAL	TABLE	23/07/2024 11:44:29
MEALORDER	TABLE	23/07/2024 18:45:54
PAKAGE	TABLE	23/07/2024 16:28:01
PRODUCT	TABLE	23/07/2024 17:06:37
SUPPLIER	TABLE	23/07/2024 17:06:49
WORKERORDER	TABLE	23/07/2024 18:44:29
WORKERS	TABLE	23/07/2024 11:44:29

Oracle Export SQL Inserts PL/SQL Developer Log

☐ Drop tables ☒ Create tables ☐ Truncate tables ☒ Delete records ☒ Disable triggers

☒ Disable foreign key constraint ☒ Include storage ☒ Include privileges

Commit every 100 records (0 = never)

Where clause

Output file C:\Users\IMOE001\minipDataBases\4\שלב 4\backup4.sql

debbi1@XE Exporting... Done