

תוכן

2.....	שלב א
2.....	תיאור המערכת
2.....	ישויות בסיס הנתונים:
2.....	תיאור מילולי של טבלאות המערכת:
3.....	דיאגרמת ERD:
4.....	תרשים DSD:
5.....	מילון מונחים:
6.....	יצירת הטבלאות (Create Table):
9.....	הזנת נתונים לטבלאות והצגתם (insert and select):
14	עדכון הטבלאות (update):
16	מחיקת רשומות (delete):
18	מחיקת עמודות / טבלאות (drop):
21	שלב ב
21	יצירת ישויות (שורות) בטבלאות:
21	יצירת קבצי csv ע"י Mockaroo:
22	טעינת קבצי csv ל plsql:
26	יצירת ישויות ע"י data generator (plsql):
40	גיבוי ואחזור הנתונים:
40	גיבוי הנתונים:
41	אחזור המידע:
44	שאלות SQL:
52	אינדקסינג:
52	טבלת השוואה לפני – אחרי האינדקסים:
53	אינדקסים יעילים:
57	אינדקסים לא-יעילים:

שלב א

תיאור המערכת

בפרויקט זה בחרנו לממש בסיס נתונים עבור גמ"ח כספים באופן גנרי.

גמ"ח הכספים בנוי באופן הבא:

אדם שמעוניין לקבל סכום כסף מהגמ"ח (להלן "הלוואה") צריך לפנות לגמ"ח הכספים (להלן "המלווה"), המלווה קובע עם הלווה את תנאי ההלוואה – סכום, תשלומים, מועד תחילת הפירעון, מועד סיום הפירעון ואמצעי התשלום.

בנוסף על הלווה להחתים לפחות **ערב** אחד על מסמך ההלוואה.

המלווה יאמת את פרטי הבנק של הלווה ולאחר מכן יבקש אישור וחתימה **מנשיא הגמ"ח**, ולאחר אישור סופי זה הגמ"ח ינפיק צ'ק בסכום שסוכם עבור הלווה.

ישויות בסיס הנתונים:

- מלווה – Lender
- הלוואה – Loan
- תשלום – Payment
- חשבון הבנק – BankAccount
- בנק – Bank
- אדם – Person
- נשיא הגמ"ח – President
- ערב – Guarantee
- לווה – Loaner

תיאור מילולי של טבלאות המערכת:

Lender (lenderBN, lenderName, lenderAddress, lenderPhone, LenderMail)

Loan (loanID, payCode, loanDate, PID, lenderBN)

Payment (payCode, dueDate, startDate, totalAmount, methodOfPayment, PID)

BankAccount (accountNumber, bankBN, balance, PID)

Bank (bankBN, bankName)

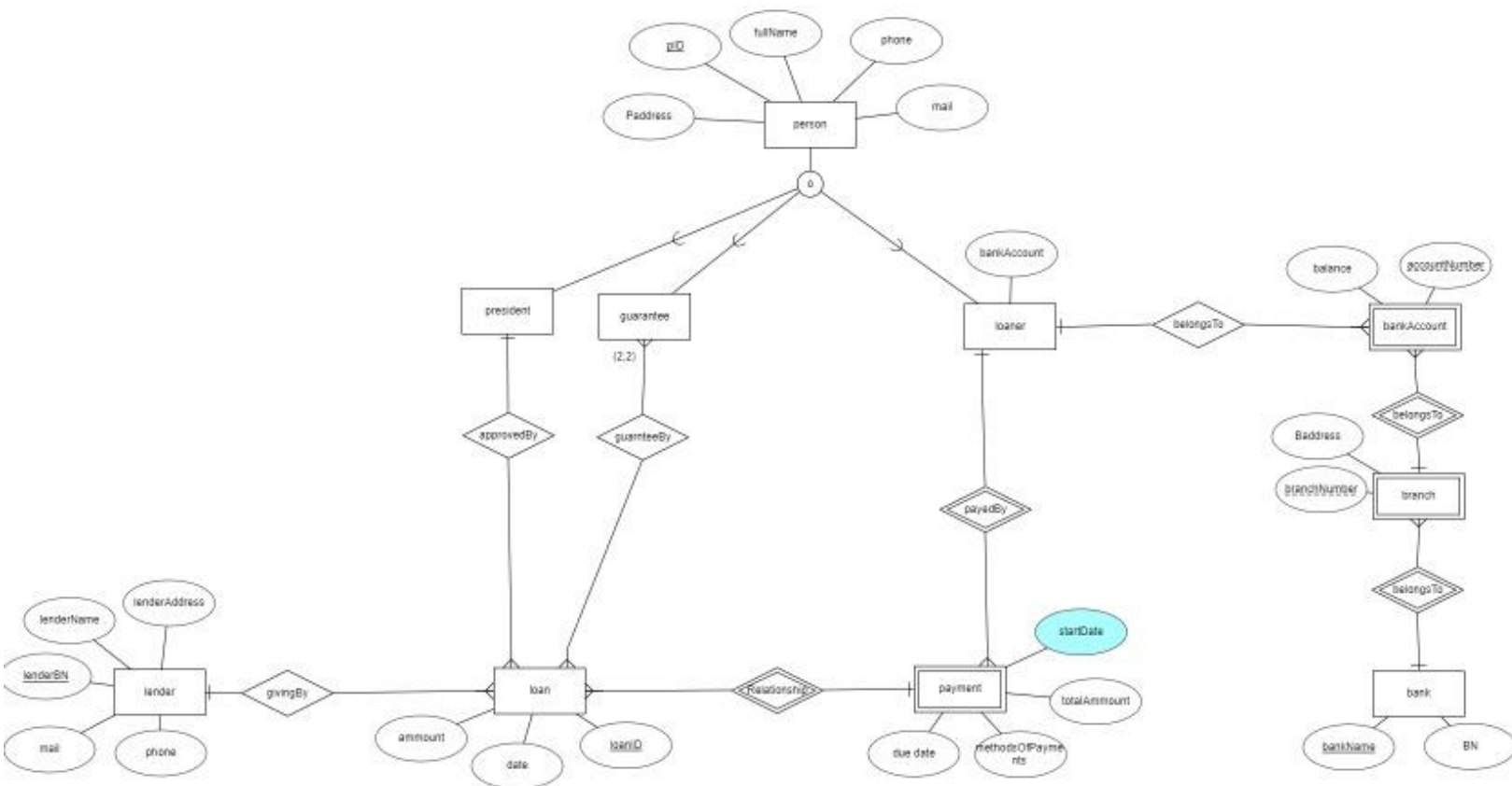
Person (PID, PfullName, Paddress, Pphone, Pmail)

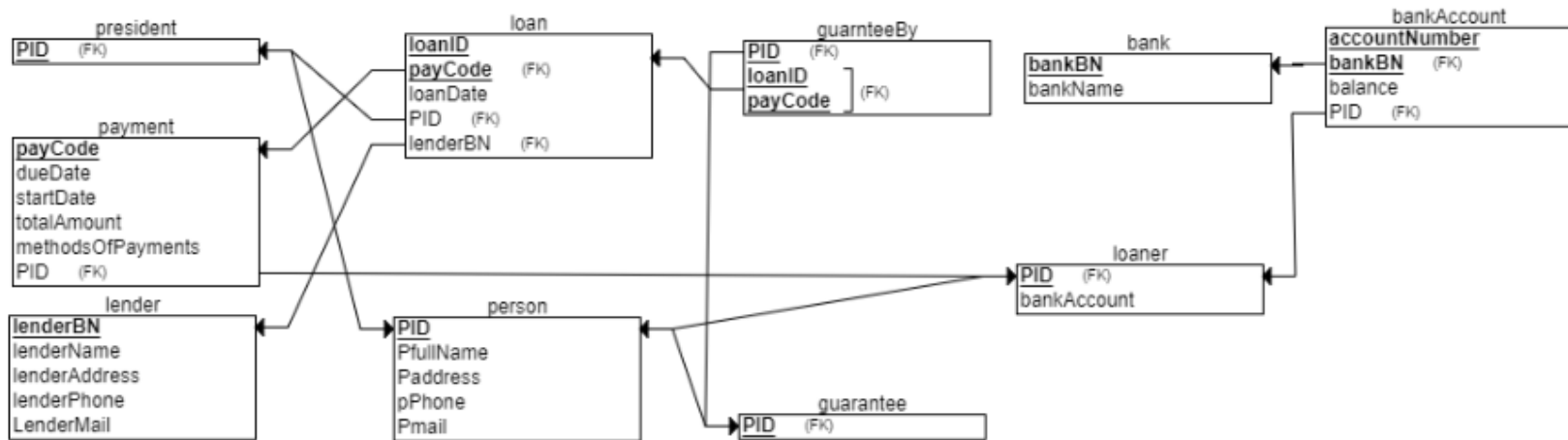
President (PID)

Guarantee (PID)

Loaner(PID, bankAccount)

GuaranteeBy (PID, loanID, payCode)





מילון מונחים:

Table name	Attribute	Explanenation
Lender	<u>lenderName</u>	שם המלווה
	<u>lenderAddress</u>	כתובת המלווה
	<u>lenderPhone</u>	טלפון המלווה
	<u>LenderMail</u>	כתובת מייל של המלווה
	<u>lenderBN</u>	מספר העסק של הגמ"ח
Person	<u>PfullName</u>	שם מלא
	<u>PID</u>	תעודת זהות
	<u>PAddress</u>	כתובת
	<u>Pphone</u>	טלפון
	<u>Pmail</u>	כתובת מייל
Persident	<u>PID</u>	תעודת זהות של הנשיא
Guarantee	<u>PID</u>	תעודת זהות של הערב
Loaner	<u>PID</u>	תעודת זהות של הלווה
	<u>BankAccount</u>	מספר חשבון בנק של הלווה
GuaranteeBy	<u>PID</u>	תעודת זהות של הערב
	<u>LoanID</u>	מספר מזהה להלוואה
	<u>payCode</u>	מספר מזהה לעסקה
Bank	<u>BankBN</u>	מספר מזהה של הבנק (ח.פ)
	<u>BankName</u>	שם הבנק
BankAccount	<u>AccountNumber</u>	מספר חשבון הבנק
	<u>BankBN</u>	מספר מזהה של הבנק
	<u>balance</u>	יתרה בחשבון
	<u>PID</u>	תעודת זהות של בעל החשבון
payment	<u>payCode</u>	מספר מזהה לעסקה
	<u>dueDate</u>	תאריך פירעון סופי להלוואה

	startDate	תחילת תשלום ההלוואה
	totalAmount	סכום ההלוואה סך הכל
	methodOfPayment	שיטת תשלום (צ'ק, אשראי, מזומן)
	Pid	תעודת זהות של הלווה
Loan	loanID	מספר מזהה להלוואה
	payCode	מספר מזהה לעסקה
	loanDate	תאריך קבלת ההלוואה
	PID	תעודת זהות של הנשיא
	LenderBN	מספר מזהה של הגמ"ח

יצירת הטבלאות (Create Table):

:Lender Table

```
CREATE TABLE lender
(
  lenderName VARCHAR(40) NOT NULL,
  lenderAddress VARCHAR(40) NOT NULL,
  lenderPhone VARCHAR(40) NOT NULL,
  LenderMail VARCHAR(40) NOT NULL,
  lenderBN INT NOT NULL,
  PRIMARY KEY (lenderBN)
);
```

:Person Table

```
CREATE TABLE person
(
  PfullName VARCHAR(40) NOT NULL,
  PID INT NOT NULL,
  Paddress VARCHAR(40) NOT NULL,
  pPhone VARCHAR(40) NOT NULL,
  Pmail VARCHAR(40) NOT NULL,
  PRIMARY KEY (PID)
);
```

:Loaner Table

```
CREATE TABLE loaner
(
  bankAccount INT NOT NULL,
  PID INT NOT NULL,
  PRIMARY KEY (PID),
  FOREIGN KEY (PID) REFERENCES person(PID)
);
```

:President Table

```
CREATE TABLE president
(
  PID INT NOT NULL,
  PRIMARY KEY (PID),
  FOREIGN KEY (PID) REFERENCES person(PID)
);
```

:Guarantee Table

```
CREATE TABLE guarantee
(
  PID INT NOT NULL,
  PRIMARY KEY (PID),
  FOREIGN KEY (PID) REFERENCES person(PID)
);
```

:Bank Table

```
CREATE TABLE bank
(
  bankName VARCHAR(40) NOT NULL,
  bankBN INT NOT NULL,
  PRIMARY KEY (bankBN)
);
```

:Payment Table

```
CREATE TABLE payment
(
  dueDate DATE NOT NULL,
  startDate DATE NOT NULL,
  totalAmount INT NOT NULL,
  methodsOfPayments VARCHAR(40) NOT NULL,
  payCode INT NOT NULL,
  PID INT NOT NULL,
  PRIMARY KEY (payCode),
  FOREIGN KEY (PID) REFERENCES loaner(PID)
);
```

:Loan Table

```
CREATE TABLE loan
(
  loanDate DATE NOT NULL,
  loanID INT NOT NULL,
  payCode INT NOT NULL,
  PID INT NOT NULL,
  lenderBN INT NOT NULL,
  PRIMARY KEY (loanID, payCode),
  FOREIGN KEY (payCode) REFERENCES payment(payCode),
);
```

```
FOREIGN KEY (PID) REFERENCES president(PID),  
FOREIGN KEY (lenderBN) REFERENCES lender(lenderBN)  
);
```

:Bank Account Table

```
CREATE TABLE bankAccount  
(  
    accountNumber INT NOT NULL,  
    balance INT NOT NULL,  
    bankBN INT NOT NULL,  
    PID INT NOT NULL,  
    PRIMARY KEY (accountNumber, bankBN),  
    FOREIGN KEY (bankBN) REFERENCES bank(bankBN),  
    FOREIGN KEY (PID) REFERENCES loaner(PID)  
);
```

:GuaranteeBy Table

```
CREATE TABLE guaranteeBy  
(  
    PID INT NOT NULL,  
    loanID INT NOT NULL,  
    payCode INT NOT NULL,  
    PRIMARY KEY (PID, loanID, payCode),  
    FOREIGN KEY (PID) REFERENCES guarantee(PID),  
    FOREIGN KEY (loanID, payCode) REFERENCES loan(loanID, payCode)  
);
```


הזנת נתונים לטבלאות והצגתם (insert and select):

Lender Table

SQL

Output

Statistics
















INSERT INTO lender(lenderBN, lenderName ,lenderAddress,lenderphone, lenderMail)
VALUES (333,'david','nachum 10','0525665512','david@gmail.com');

INSERT INTO lender(lenderBN, lenderName ,lenderAddress,lenderphone, lenderMail)
VALUES (444,'levi','nachum 11','0525665512','leci@gmail.com');

INSERT INTO lender(lenderBN, lenderName ,lenderAddress,lenderphone, lenderMail)
VALUES (55,'shlomo','nachum 12','0525665514','david@gmail.com');

INSERT INTO lender(lenderBN, lenderName ,lenderAddress,lenderphone, lenderMail)
VALUES (66,'ori','nachum 13','0525665514','ori@gmail.com');

|
commit;

	LENDERNAME	LENDERADDRESS	LENDERPHONE	LENDERMAIL	LENDERBN
1	Daniel Ventura	nachum 9	541234567	daniel@gmail.com	111
2	david	nachum 10	0525665512	david@gmail.com	333
3	levi	nachum 11	0525665512	leci@gmail.com	444
4	shlomo	nachum 12	0525665514	david@gmail.com	55
5	ori	nachum 13	0525665514	ori@gmail.com	66

Person Table

SQL

Output

Statistics


```
INSERT INTO person(PID, pfullname, paddress, pphone, pmail)
VALUES (1111,'david david','batata 15','0500000000','david@gmail.com')

INSERT INTO person(PID, pfullname, paddress, pphone, pmail)
VALUES (2222,'dodo dodo','batata 16','0511111111','dodo@gmail.com');

INSERT INTO person(PID, pfullname, paddress, pphone, pmail)
VALUES (3333,'moshe moshe','batata 17','0522222222','moshe@gmail.com')

commit;

select * from person;
```



	PFULLNAME	PID	PADDRESS	PPHONE	PMAIL
1	david david	1111	batata 15	0500000000	david@gmail.com
2	dodo dodo	2222	batata 16	0511111111	dodo@gmail.com
3	moshe moshe	3333	batata 17	0522222222	moshe@gmail.com

:President Table

SQL

Output

Statistics

```

INSERT INTO president(pid)
VALUES (1111);

INSERT INTO president(pid)
VALUES (2222);

commit;

SeLECT * FROM president;
    
```

	PID
1	1111
2	2222

:Guarantee Table

SQL

Output

Statistics

```

INSERT INTO guarantee(pid)
VALUES (2222);
|
commit;

select * from guarantee;
    
```

















Insert guarantee

Commit

Select guarantee

:Bank Account Table

SQL	Output	Statistics
<pre>INSERT INTO bankaccount(accountnumber, balance, bankbn, pid) VALUES (2345, -9999, 33, 3333); INSERT INTO bankaccount(accountnumber, balance, bankbn, pid) VALUES (1234, 3333, 20, 3333); commit; select * from bankaccount;</pre>		

Insert bankaccount		Commit		Select bankaccount	
					
					
					
	ACCOUNTNUMBER	BALANCE	BANKBN	PID	
▶ 1	2345	-9999	33	3333	
2	1234	3333	20	3333	

:Payment Table

SQL


















Output

Statistics

```
INSERT INTO payment(duedate, startdate, totalamount, methodsofpayments, paycode, pid)
VALUES (sysdate, sysdate -100, 777, 'credit',999,3333);

commit;

SeLECT * FROM payment;
```

	DUEDATE	STARTDATE	TOTALAMOUNT	METHODSOFPAYMENTS	PAYCODE	PID
1	08/03/2022 10:55:10	28/11/2021 10:55:10	777	credit	999	3333

:Loan Table

SQL Output Statistics

```
INSERT INTO loan(loandate, loanid, paycode, pid, lenderbn)
VALUES (sysdate-150, 789, 999,1111, 444);

commit;
|
SeLECT * FROM loan;
```

	LOANDATE	LOANID	PAYCODE	PID	LENDERBN
1	09/10/2021 10:58:48	789	999	1111	444

:Guarantee By Table

SQL Output Statistics

```
INSERT INTO guaranteeBy(Pid, Loanid, Paycode)
VALUES (2222, 789, 999);

commit;
|
SeLECT * FROM guaranteeBy;
```

	PID	LOANID	PAYCODE
1	2222	789	999

עדכון הטבלאות (update):

Lender Table:

SQL Output Statistics

```

update lender
set lenderPhone=0523456789
where lenderMail='david@gmail.com';

commit;

SeLECT * FROM lender;
    
```

Update lender Commit Select lender

	LENDERNAME	LENDERADDRESS	LENDERPHONE	LENDERMAIL	LENDERBN
1	Daniel Ventura	nachum 9	541234567	daniel@gmail.com	111
2	david	nachum 10	523456789	david@gmail.com	333
3	levi	nachum 11	0525665512	leci@gmail.com	444
4	shlomo	nachum 12	523456789	david@gmail.com	55
5	ori	nachum 13	0525665514	ori@gmail.com	66

Payment Table:

SQL Output Statistics

```

update payment
set totalAmount=5
where paycode=999;

commit;

SeLECT * FROM payment;
    
```

Update payment Commit Select payment

	DUEDATE	STARTDATE	TOTALAMOUNT	METHODSOFPAYMENTS	PAYCODE	PID
1	08/03/2022 10:55:10	28/11/2021 10:55:10	5	credit	999	3333

SQL Output Statistics

```

update payment
set totalAmount=20
where totalAmount=5;

commit;

SELECT * FROM payment;
    
```

Update payment Commit Select payment

	DUE DATE	START DATE	TOTAL AMOUNT	METHODS OF PAYMENTS	PAY CODE	PID
1	08/03/2022 10:55:10	28/11/2021 10:55:10	20	credit	999	3333

:Bank Account Table

SQL Output Statistics

```

update bankaccount
set balance=123456
where accountNumber=2345;

commit;

SELECT * FROM bankaccount;
    
```

Update bankaccount Commit Select bankaccount

	ACCOUNT NUMBER	BALANCE	BANK BN	PID
1	2345	123456	33	3333
2	1234	3333	20	3333

מחיקת רשומות (delete):

Lender Table:

SQL Output Statistics

```
delete FROM lender WHERE lenderAddress='nachum 10';
```

```
SeLECT * FROM lender;
```

	LENDERNAME	LENDERADDRESS	LENDERPHONE	LENDERMAIL	LENDERBN
1	Daniel Ventura	nachum 9	541234567	daniel@gmail.com	111
2	levi	nachum 11	0525665512	leci@gmail.com	444
3	shlomo	nachum 12	523456789	david@gmail.com	55
4	ori	nachum 13	0525665514	ori@gmail.com	66

SQL Output Statistics

```
delete FROM lender WHERE lenderBN=66;
```

```
SeLECT * FROM lender;
```

Delete lender Select lender

	LENDERNAME	LENDERADDRESS	LENDERPHONE	LENDERMAIL	LENDERBN
1	Daniel Ventura	nachum 9	541234567	daniel@gmail.com	111
2	levi	nachum 11	0525665512	leci@gmail.com	444
3	shlomo	nachum 12	523456789	david@gmail.com	55

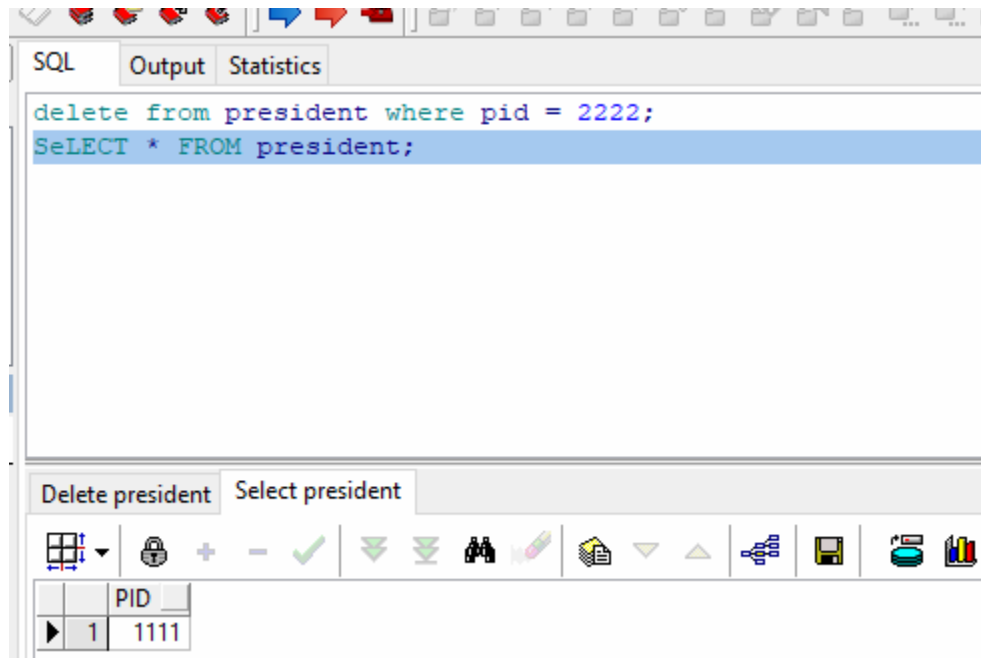
SQL Output Statistics

```
delete FROM lender WHERE lenderBN=111;
```

```
SeLECT * FROM lender;
```

Delete lender Select lender

	LENDERNAME	LENDERADDRESS	LENDERPHONE	LENDERMAIL	LENDERBN
1	levi	nachum 11	0525665512	leci@gmail.com	444
2	shlomo	nachum 12	523456789	david@gmail.com	55



:Payment Table

SQL Output Statistics

```
ALTER TABLE payment
drop COLUMN dueDate;

SELECT * FROM payment;
```

Alter payment Select payment

	STARTDATE	TOTALAMOUNT	METHODSOFPAYMENTS	PAYCODE	PID
1	28/11/2021 10:55:10	20	credit	999	3333

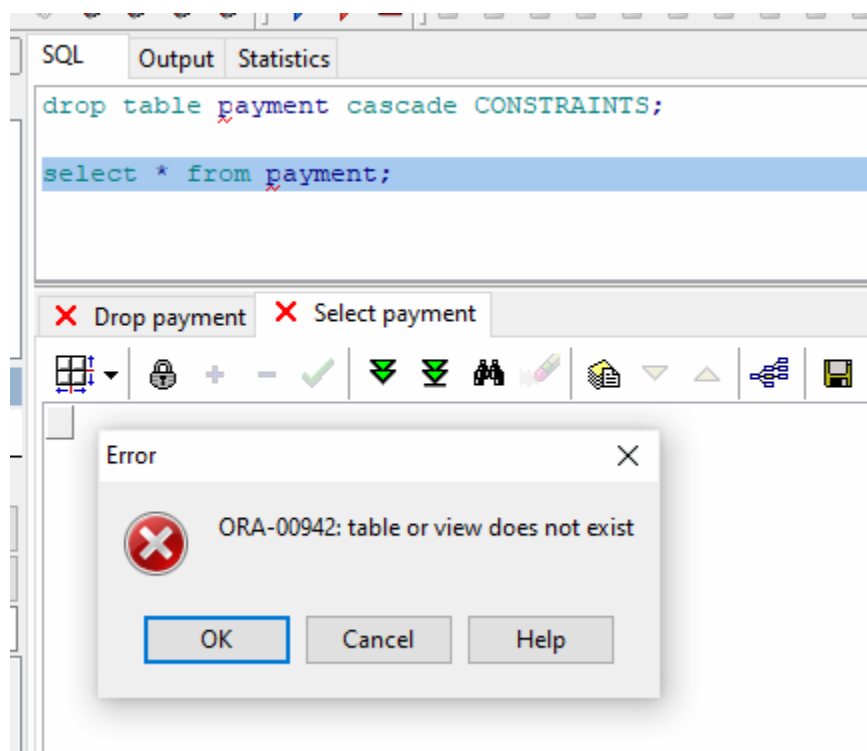
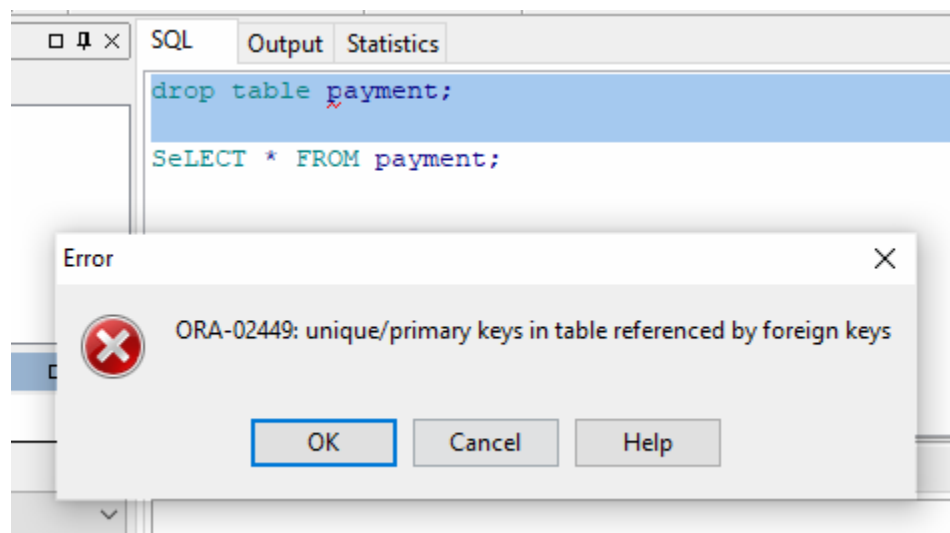
SQL Output Statistics

```
ALTER TABLE payment
drop COLUMN startDate;
```

Alter payment Select payment

	TOTALAMOUNT	METHODSOFPAYMENTS	PAYCODE	PID
1	20	credit	999	3333

כשניסינו למחוק את payment table קיבלנו הודעת שגיאה שלא ניתן למחוק את הטבלה כיון שיש לה בנים לכן נדרשנו להשתמש בפקודה "cascade" בכדי לבצע מחיקה "עמוקה".



ואכן ניתן להסיק שהטבלה נחקה כיון שבפקודת select לא ניתן למצוא אותה "does not exist"

:Person Table

The screenshot shows a database management interface with two main panes. The top pane, titled 'SQL', contains the following SQL commands:

```
ALTER TABLE person  
drop COLUMN pmail;  
  
SeLECT * FROM person;
```

The bottom pane, titled 'Select person', displays the result of the SQL query as a table. The table has five columns: an index, PFULLNAME, PID, PADDRESS, and PPHONE. The data is as follows:

	PFULLNAME	PID	PADDRESS	PPHONE
1	david david	1111	batata 15	0500000000
2	dodo dodo	2222	batata 16	0511111111
3	moshe moshe	3333	batata 17	0522222222

שלב ב

יצירת ישויות (שורות) בטבלאות:

יצירת קבצי csv ע"י Mockaroo:

עבור ישות הבנק נחולל מידע עבור שתי השדות שלו – bankBN, bankName.
נחולל 100 שורות וניצור קובץ csv שיכיל את הנתונים (בהמשך נטען אותו לsql pl)

Field Name	Type	Options
bankBN	Row Number	blank: 0 % Σ X
bankName	Fake Company Name	blank: 0 % Σ X

ADD ANOTHER FIELD

Rows: 100 Format: CSV Line Ending: Unix (LF) Include: ☐ header ☐ BOM

עבור ישות lender נחולל מידע עבור כל השדות שלו כך שכל שדה יאותחל במידע בהתאם לדרישה (כתובת, שם מלא וכו')

נחולל את הנתונים וניצור קובץ csv שיכיל את הנתונים (בהמשך נטען אותו לsql/pl)

Field Name	Type	Options
lenderAddress	Street Address	blank: 0 % Σ X
lenderName	Full Name	blank: 0 % Σ X
lenderBN	Row Number	blank: 0 % Σ X
lenderMail	Email Address	blank: 0 % Σ X
lenderPhone	Phone	format: ##### blank: 0 % Σ X

ADD ANOTHER FIELD

Rows: 500 Format: CSV Line Ending: Windows (CRLF) Include: ☐ header ☐ BOM

טעינת קבצי csv ל plsql:

כעת נטען את קבצי csv שיצרנו ב Mockaroo ל pl/sql.

כיון שקבצי csv נוצרו עם headers המערכת מזהה את השדות של הישויות (במרכז בצד שמאל).

נטען את הקבצים של lender ושל bank.

לאחר טעינת הקבצים נבצע שאילתה להצגת הנתונים ואכן כל הישויות יוצגו כמצופה.

Data from Textfile | Data to Oracle

General

Owner: SYSTEM | Table: LENDER | ☐ Clear Table

Commit every...: 100 | ☒ Overwrite duplicates | ☐ Ignore duplicates

Initializing Script: ...

Finalizing Script: ...

Fields

Field	Field
Field1 lenderAddress -> LENDERADDRESS	Field LENDERPHONE
Field2 lenderName -> LENDERNNAME	Fieldtype String
Field3 lenderBN -> LENDERBN	<input type="button" value="Create SQL"/>
Field4 LenderMail -> LENDERMAIL	SQL function <input type="text"/> ...
Field5 lenderPhone -> LENDERPHONE	additional Oracle processing, for example: substr(%, 1, 20)

Result Preview

lenderAddress	lenderName	lenderBN	LenderMail	lenderPhone
33 Cherokee Street	Dinnie Casine	1	dcasine0@apache.org	4197484951
249 Hanson Parkway	Garnette Rominov	2	grominov1@chronoengine.com	9026233387

SYSTEM@XE 500 records imported in 1.079 seconds

צור איתן לוי

SQL Output Statistics					
select * from lender where lenderbn > 480;					
	LENDERNAME	LENDERADDRESS	LENDERPHONE	LENDERMAIL	LENDERBN
1	Marchelle Karchowski	24882 Russell Junction	8935774343	mkarchowskidc@dmoz.org	481
2	Bogey Domeny	8288 Golf Course Court	7478162103	bdomenydd@google.ru	482
3	Obidiah Fordyce	742 Rutledge Point	7829577449	ofordyce@toplist.cz	483
4	Natalee Windybank	4 Little Fleur Alley	5688857848	nwindybankdf@twitter.com	484
5	Maxwell Hathaway	0 Burning Wood Hill	9058031861	mhathawaydg@washingtonpost.com	485
6	Jeffry Macklin	3703 Scott Pass	7866367499	jmacklin@redcross.org	486
7	Reine Jerrard	94 Sage Point	1187698439	rjerrarddi@ebay.co.uk	487
8	Merwin Hayland	848 4th Road	6895227453	mhaylanddj@go.com	488
9	Myrilla Leadbitter	81 Knutson Way	7431598195	mleadbitterdk@howstuffworks.com	489
10	Briggs Getch	104 Claremont Park	3477164221	bgetchdl@jugem.jp	490
11	Sharona Cristofari	56 Longview Terrace	2947984295	scristofaridm@nasa.gov	491
12	Geno Growy	6141 Westerfield Lane	2851165065	ggrowydn@marriott.com	492
13	Hillery O'Carrol	0146 Tony Street	4284632866	hocarroldo@behance.net	493
14	Zachariah Griswood	4027 Dovetail Pass	3606964550	zgriswooddp@wordpress.org	494
15	Crista Lugton	54 Novick Trail	7768847744	clugtondq@usa.gov	495
16	Estelle Calderon	404 Moulton Alley	1507968097	ecalderondr@godaddy.com	496
17	Waly Kleisle	3131 International Street	9862108781	wkleisleds@cocolog-nifty.com	497
18	Emory Wedlake	0 Packers Drive	3212246736	ewedlakedt@networkadvertising.org	498
19	Jordan Izakov	98294 Golden Leaf Park	3248313111	jizakovdu@scientificamerican.com	499
20	Randal Bedburrow	1253 Mosinee Junction	4442831764	rbedburrowdv@biblegateway.com	500

1:42 SYSTEM@XE 20 rows selected in 0.047 seconds

Data from Textfile

Data to Oracle

General

Owner

SYSTEM

Table

BANK

Clear Table

Commit every...

100

Overwrite duplicates

Ignore duplicates

Initializing Script

Finalizing Script

Fields

Field1 bankBN -> BANKBN

Field2 bankName -> BANKNAME

Field

BANKNAME

Fieldtype

String

Create SQL

SQL function

additional Oracle processing, for example: substr(#, 1, 20)

Result Preview

bankBN	bankName
1	Zathin
2	Alphazap

Import

Import to Script

Close

SYSTEM@XE

500 records imported in 0.453 seconds

Help

יצירת ישויות ע"י (plsql) data generator:

נניצר ישויות עבור טבלת Person.

ניצור 20,000 ישויות ונזין בשדות מידע לפי דרישה (שם פרטי + שם משפחה, מספר בטוח מוגדר וכו').

נשתמש בתבניות ופונקציות מוכנות מראש מתוך ה data generator.

PERSON

Owner	Table	Number of records
SYSTEM	PERSON	20000

Name	Type	Size	Data
PFULLNAME	VARCHAR2	40	FirstName + LastName
PID	NUMBER		Sequence(10000, 1, 99999)
PADDRESS	VARCHAR2	40	Address1
PPHONE	VARCHAR2	40	'05' + random(20000000, 99999999)
PMAIL	VARCHAR2	40	Email
*			

Definition Options Result

SYSTEM@XE 20000 records generated in 12.969 seconds














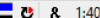

קעת נציג את הנתונים מטבלת person.

SQL


Output

Statistics

select * from person where pid > 29980;

	PFULLNAME	PID	PADDRESS	PPHONE	PMAIL
1	StockardWahlberg	29981	50 Lara Road	0592933647	stockardw@httprint.uk
2	JuliannaCheadle	29982	52 Marsden Ave	0590650221	jcheadle@ach.gr
3	PercyAdkins	29983	503 Foley	0541110526	percy.adkins@nbs.jp
4	MaceoReeves	29984	40 Miki Street	0560172501	maceo.reeves@diamondgroup.de
5	DebiHarrison	29985	7 Brendan Road	0556847644	debi@bluffcitysteel.in
6	VincentAkins	29986	12 Webb Blvd	0582050645	vincent.akins@columbiabancorp.jp
7	AlbertBrickell	29987	82 Colman Street	0597991805	albertb@gulfmarkoffshore.jp
8	DavySpacek	29988	18 Zellweger Street	0589538207	davy.s@oss.jp
9	SamanthaPorter	29989	61 Mel Blvd	0549422423	samantha.porter@randomwalk.com
10	JosephRooker	29990	683 Baldwin Drive	0537867655	j.rooker@its.com
11	JenaAtlas	29991	2 Berwyn Ave	0533809482	jena.atlas@jsa.com
12	JonJoli	29992	1 Whitman	0541846593	jon.joli@sps.com
13	ScarlettCrow	29993	98 Zeta-Jones Road	0588008981	scarlett@connected.jp
14	GeoffreyZellweger	29994	7 Weisz Street	0540213667	geoffrey.zellweger@pscfogroup.ch
15	TreatGoldblum	29995	34 James Road	0578681350	treat.goldblum@vspan.com
16	FredaHamilton	29996	56 Black Blvd	0583425268	freda.hamilton@pragmatechsoftware.dk
17	AidaBacon	29997	70 Melbourne Drive	0540022582	a.bacon@mss.de
18	KevnPhillips	29998	30 Nakai Street	0562342272	k.phillips@capstone.uk
19	HenryTobolowsky	29999	33 Wagner Street	0552553264	henry.tobolowsky@atlanticcredit.ar

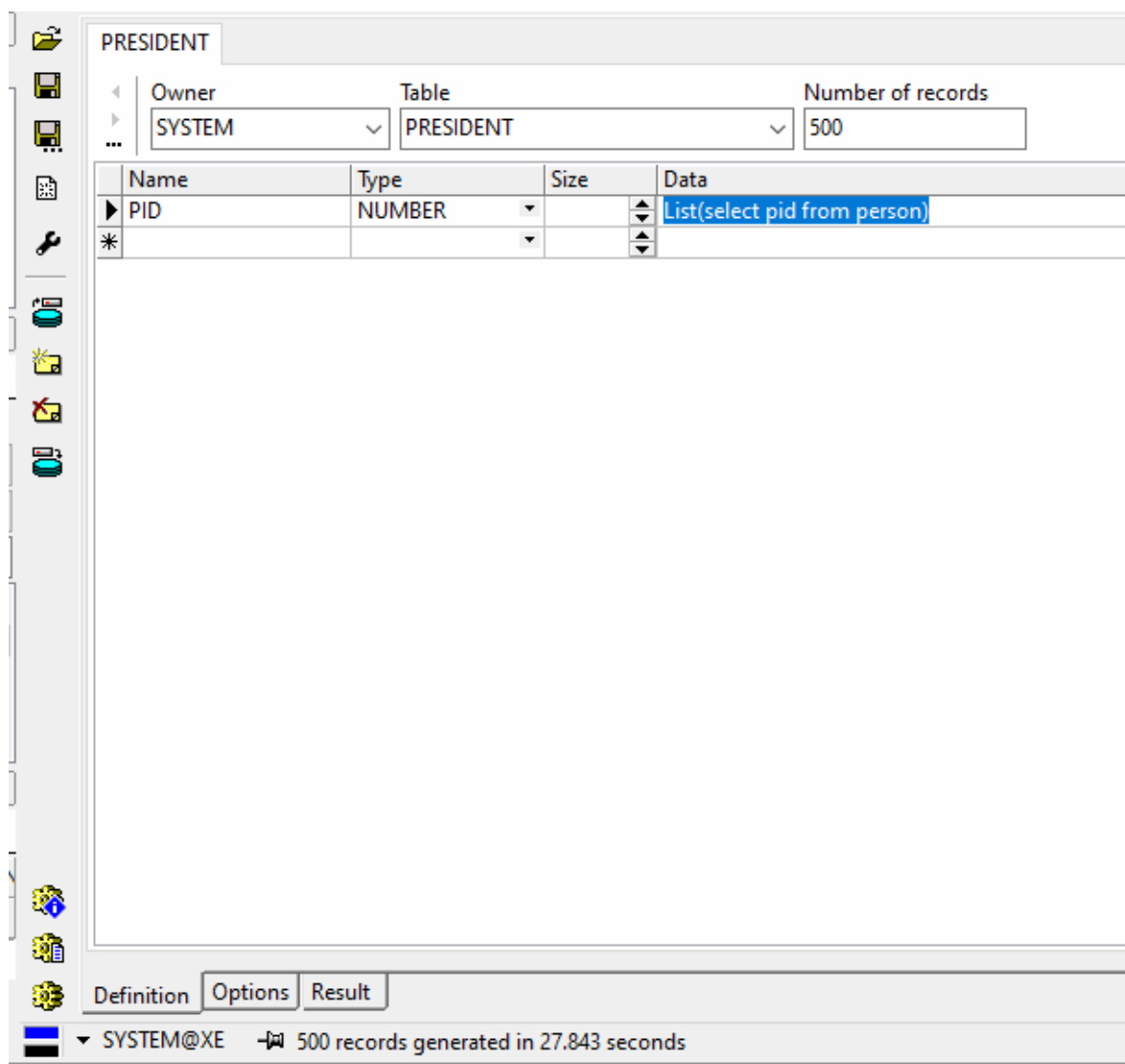
 1:40 SYSTEM@XE 19 rows selected in 0.062 seconds

נניצר ישויות עבור טבלת President.

ניצור 500 ישויות ונזין בשדות מידע לפי דרישה (במקרה הנ"ל כיון שיש ירושה מ Person נידרש להזין שדה אחד בלבד).

נשתמש בתבניות ופונקציות מוכנות מראש מתוך ה data generator.

כיון שישנה ירושה נשתמש בפונקציה "List" שמאפשרת לנו "לשדרך" בין President לישות Person קיימת.



כעת נציג את הנתונים שיצרנו בעזרת שאילתה.

The screenshot shows the Oracle SQL Developer interface. The top tabs are 'SQL', 'Output', and 'Statistics'. The 'SQL' tab is active, displaying the query: `select * from president;`. Below the query editor is a toolbar with various icons for execution and formatting. The 'Output' tab is selected, showing a table with 22 rows. The table has two columns: an index (1-22) and a column named 'PID'. The status bar at the bottom indicates 'SYSTEM@XE' and '22 rows selected in 0.078 seconds (more...)'.

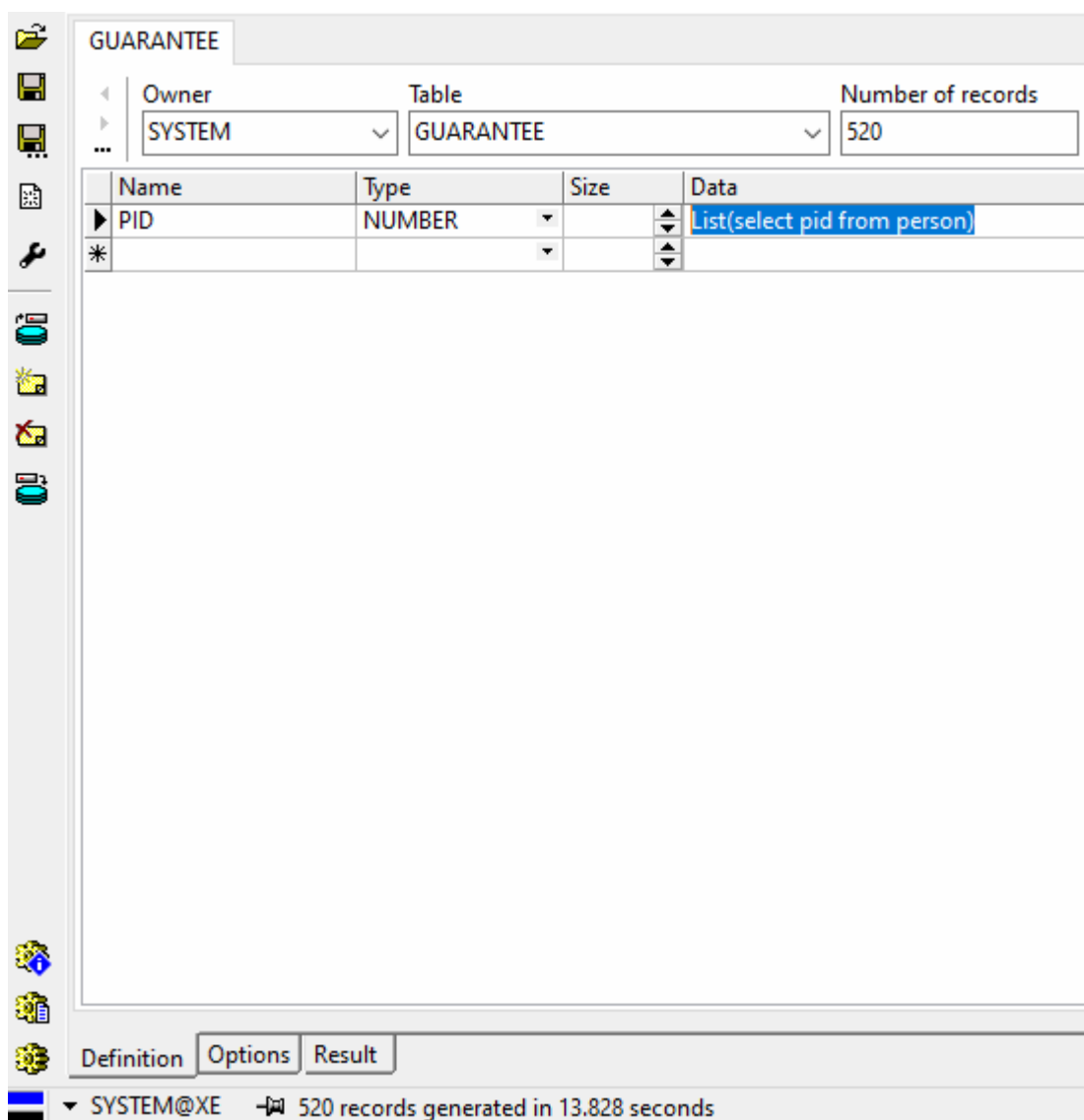
	PID
1	22659
2	21632
3	22917
4	21622
5	22342
6	15944
7	29111
8	21439
9	24080
10	29982
11	26489
12	26513
13	11605
14	14546
15	24773
16	19046
17	18878
18	29165
19	25207
20	18297
21	22403
22	12406

נייצר ישויות עבור טבלת Guarantee.

ניצור 500 ישויות (סדר גודל) ונזין בשדות מידע לפי דרישה (במקרה הנ"ל כיון שיש ירושה מ Person נדרש להזין שדה אחד בלבד).

נשתמש בתבניות ופונקציות מוכנות מראש מתוך ה data generator.

כיון שישנה ירושה נשתמש בפונקציה "List" שמאפשרת לנו "לשדרך" בין Guarantee לישות Person קיימת.



כעת בעזרת שאילתה נציג את הנתונים שבטבלת Guarantee.

SQL

Output

Statistics

```
select * from guarantee;
```

	PID
1	29948
2	19409
3	26578
4	24367
5	21142
6	19733
7	29014
8	15857
9	11207
10	13509
11	16998
12	20570
13	25143
14	26977
15	22952
16	21755
17	19837
18	22775
19	26061
20	20287
21	19437
22	15541

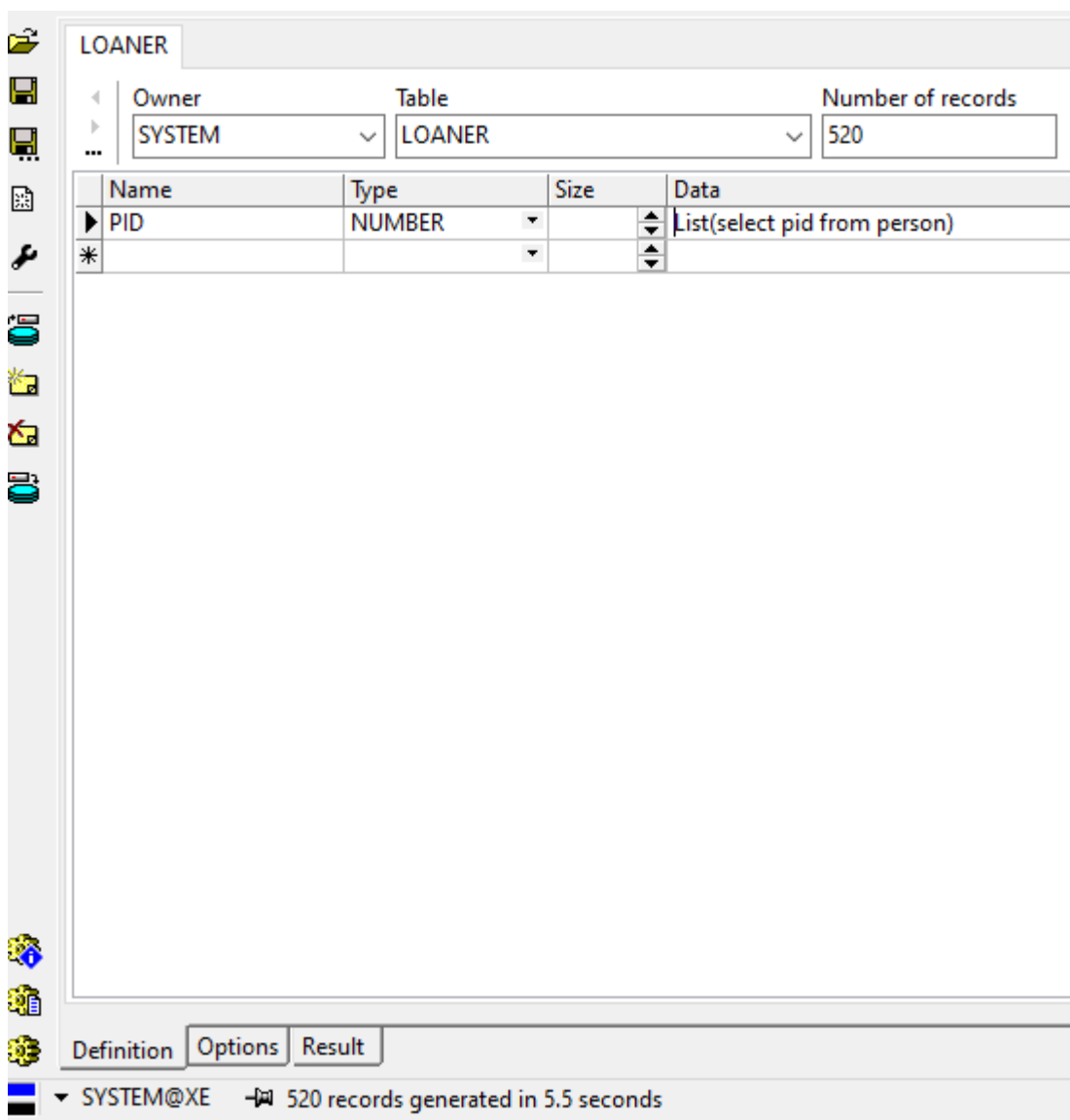
1:25
SYSTEM@XE
513 rows selected in 0.25 seconds

נייצר ישויות עבור טבלת Loaner.

ניצור 500 ישויות (סדר גודל) ונזין בשדות מידע לפי דרישה (במקרה הנ"ל כיון שיש ירושה מ Person נדרש להזין שדה אחד בלבד).

נשתמש בתבניות ופונקציות מוכנות מראש מתוך ה data generator.

כיון שישנה ירושה נשתמש בפונקציה "List" שמאפשרת לנו "לשדר" בין Loaner לישות Person קיימת.



כעת נציג את המידע מטבלת Loaner ע"י שאילתה.

SQL		Output	Statistics
		<pre>select * from loaner;</pre>	
	PID		
1	12366		
2	28422		
3	10758		
4	13302		
5	28300		
6	19462		
7	24975		
8	10316		
9	17211		
10	14198		
11	24539		
12	12471		
13	24174		
14	22532		
15	16597		
16	18664		
17	14481		
18	19904		
19	18247		
20	28873		
21	15457		
22	15119		

נייצר ישויות עבור טבלת BankAccount.

ניצור 500 ישויות (סדר גודל) ונזין בשדות מידע לפי דרישה (מספר בטווח מוגדר וכו').

נשתמש בתבניות ופונקציות מוכנות מראש מתוך ה data generator.

כיון ש BankAccount ישות חלשה ל Bank נשתמש בפונקציה "List" שמאפשרת לנו "לשדך" בין BankAccount לישות Bank קיימת.

The screenshot shows the Oracle SQL Developer Data Generator interface. The table selected is **BANKACCOUNT** from the **SYSTEM** schema. The number of records to generate is set to **600**. The columns and their data generation rules are as follows:

Name	Type	Size	Data
ACCOUNTNUMBER	NUMBER		Sequence(100, 1, 999)
BALANCE	NUMBER		Random(-50000, 100000)
BANKBN	NUMBER		List(select bankbn from bank)
*			

At the bottom, the status bar indicates: **SYSTEM@XE** 600 records generated in 0.844 seconds.

כעת ע"י שאילתה נציג את נתוני טבלת BankAccount.

		Output Statistics		
		select * from bankaccount;		
		ACCOUNTNUMBER	BALANCE	BANKBN
1		100	91038	38
2		101	42921	158
3		102	4294935668	144
4		103	7903	351
5		104	77808	53
6		105	4294945633	267
7		106	19469	113
8		107	4294942074	298
9		108	4294934621	203
10		109	48509	45
11		110	4294922634	300
12		111	4294962927	147
13		112	60970	382
14		113	25793	21
15		114	4638	95
16		115	30088	252
17		116	28438	244
18		117	80152	7
19		118	4884	279
20		119	28002	460
21		120	63962	92
22		121	13624	340

1:26 SYSTEM@XE 600 rows selected in 0.235 seconds

גמ"ח כספים

צור איתן לוי 205431935

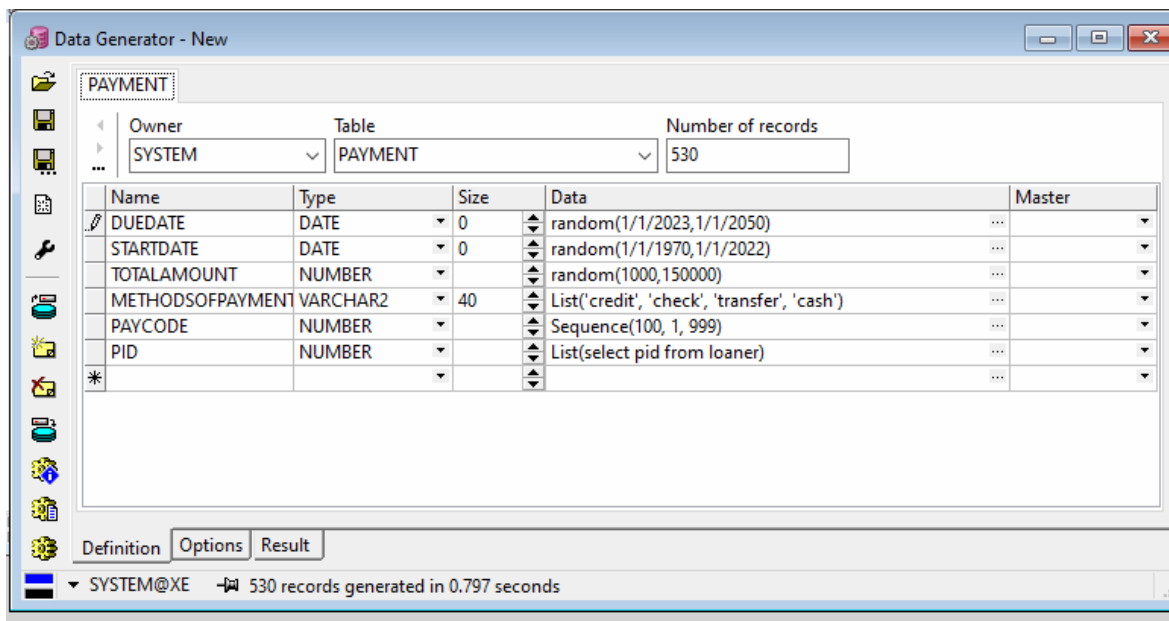
אליסף דימנט 204006415

נייצר ישויות עבור טבלת Payment.

ניצור 500 ישויות (סדר גודל) ונזין בשדות מידע לפי דרישה (מספר בטווח מוגדר, תאריך הגיוני להתחלה ולסיום וכו').

נשתמש בתבניות ופונקציות מוכנות מראש מתוך ה data generator.

כיון ש Payment מכילה ID של Loaner נשתמש בפונקציה "List" שמאפשרת לנו "לשדך" בין Payment לישות Loaner קיימת.



בעזרת שאילתה נציג את הנתונים מטבלת Payment.

SQL Window - select * from payment;

SQL Output Statistics

```
select * from payment;
```

	DUE DATE	START DATE	TOTAL AMOUNT	METHODS OF PAYMENTS	PAY CODE	PID
1	25/02/2046	08/04/2014	41652	credit	100	26060
2	26/09/2027	21/09/1999	74476	check	101	28880
3	05/01/2049	29/04/2018	89500	cash	102	20258
4	03/03/2024	04/01/2003	77959	cash	103	21187
5	01/01/2028	25/05/1999	22772	transfer	104	21522
6	18/08/2037	05/04/1990	90762	check	105	28428
7	09/07/2036	23/10/2012	121891	credit	106	27830
8	01/03/2023	29/06/1998	32039	credit	107	16278
9	30/08/2047	04/04/2015	41343	transfer	108	12583

530 rows selected in 0.328 seconds

Data Generator - New

LOAN

Owner: SYSTEM Table: LOAN Number of records: 22000

Name	Type	Size	Data	Master
LOAN DATE	DATE		List(select startDate from payment) - 30	
LOAN ID	NUMBER		Sequence(11111, 1, 99999)	
PAY CODE	NUMBER		List(select paycode from payment)	
PID	NUMBER		List(select pid from president)	
LENDER BN	NUMBER		List(select lenderbn from lender)	
*				

22000 records generated in 19.094 seconds

גמ"ח כספים

צור איתן לוי 205431935

אליסף דימנט 204006415

נייצר ישויות עבור טבלת PayedBy.

ניצור 500 ישויות (סדר גודל) ונזין בשדות מידע לפי דרישה.

נשתמש בתבניות ופונקציות מוכנות מראש מתוך ה data generator.

כיון ש PayedBy מכילה ID של Loaner ושל Payment נשתמש בפונקציה "List" שמאפשרת לנו "לשדך" בין Payment לישויות Payment I Loaner קיימות.

PAYEDBY

<	Owner	Table	Number of records
>	SYSTEM	PAYEDBY	10000
...			

Name	Type	Size	Data
PID	NUMBER		List(select pid from loaner)
PAYCODE	NUMBER		List(select paycode from payment)

DefinitionOptionsResult

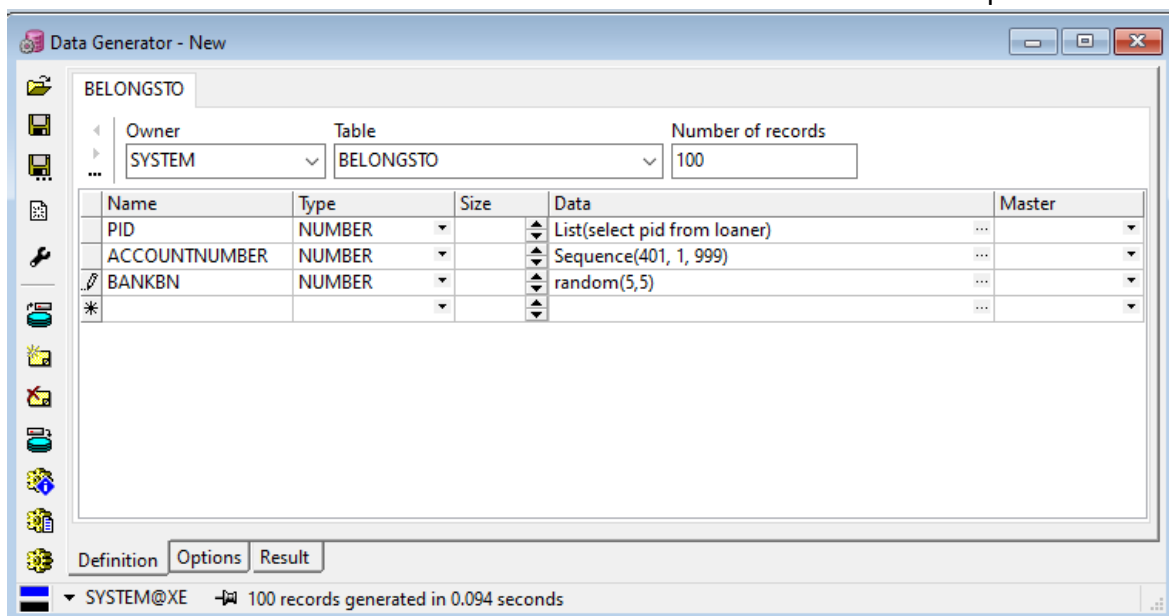
SYSTEM@XE 9729 records generated in 218.125 seconds

נייצר ישויות עבור טבלת BelongsTo.

ניצור ישויות ונזין בשדות מידע לפי דרישה.

נשתמש בתבניות ופונקציות מוכנות מראש מתוך ה data generator.

כיון ש BelongsTo מכילה ID של Loaner נשתמש בפונקציה "List" שמאפשרת לנו "לשדך" בין Payment לישויות Loaner קיימת.



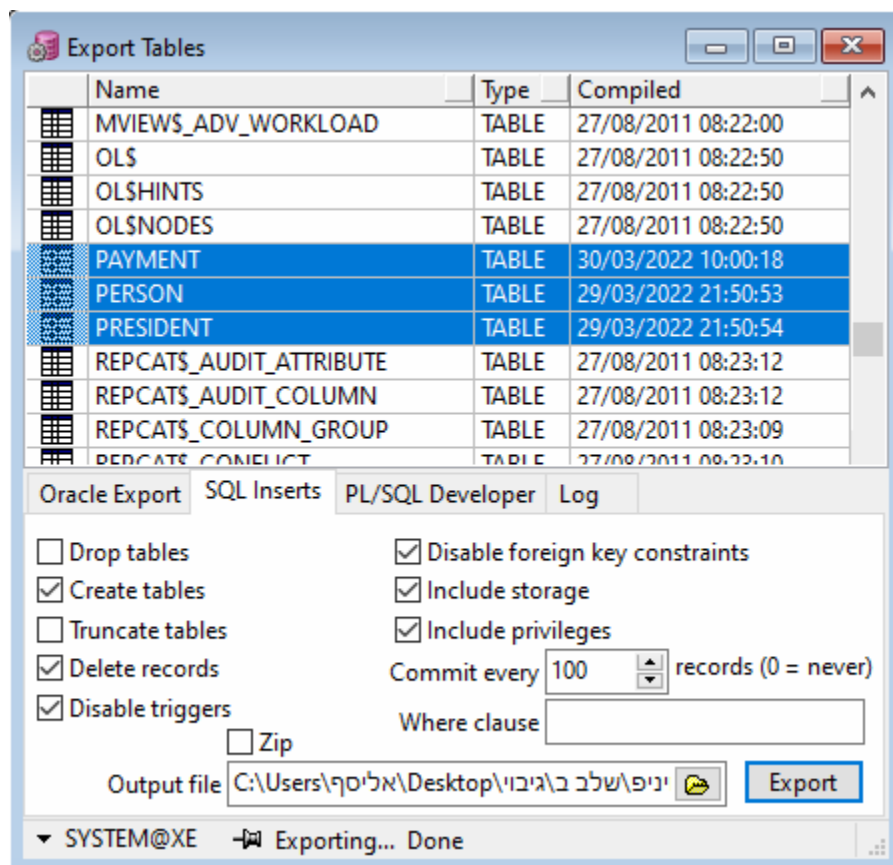
גיבוי ואחזור הנתונים:

גיבוי הנתונים:

נגבה את הנתונים דרך Export Tables -> tools -> pl/sql

נבחר את הטבלאות הרצויות לגיבוי (בחרנו את כל הטבלאות שיצרנו)

נבצע Export ל sql script.



אחזור המידע:

כעת נבצע דוגמא לשחזור נתונים ע"י שימוש בקובץ הגיבוי:

כך נראה קובץ הגיבוי – SQL למידע שייצרנו.

```

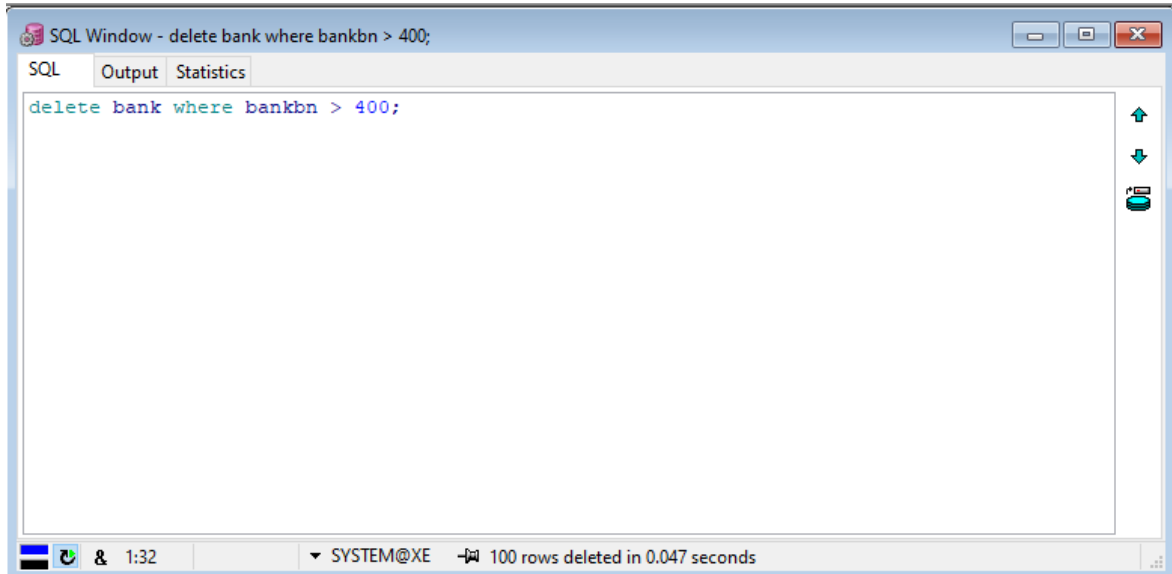
prompt PL/SQL Developer import file
prompt Created on 2022 by אליסף דימנט 04 אפריל
set feedback off
set define off
prompt Creating BANK...
create table BANK
(
  bankname VARCHAR2(40) not null,
  bankbn   INTEGER not null
)
tablespace SYSTEM
pctfree 10
pctused 40
initrans 1
maxtrans 255
storage
(

```

נציג את המידע בטבלת Bank:

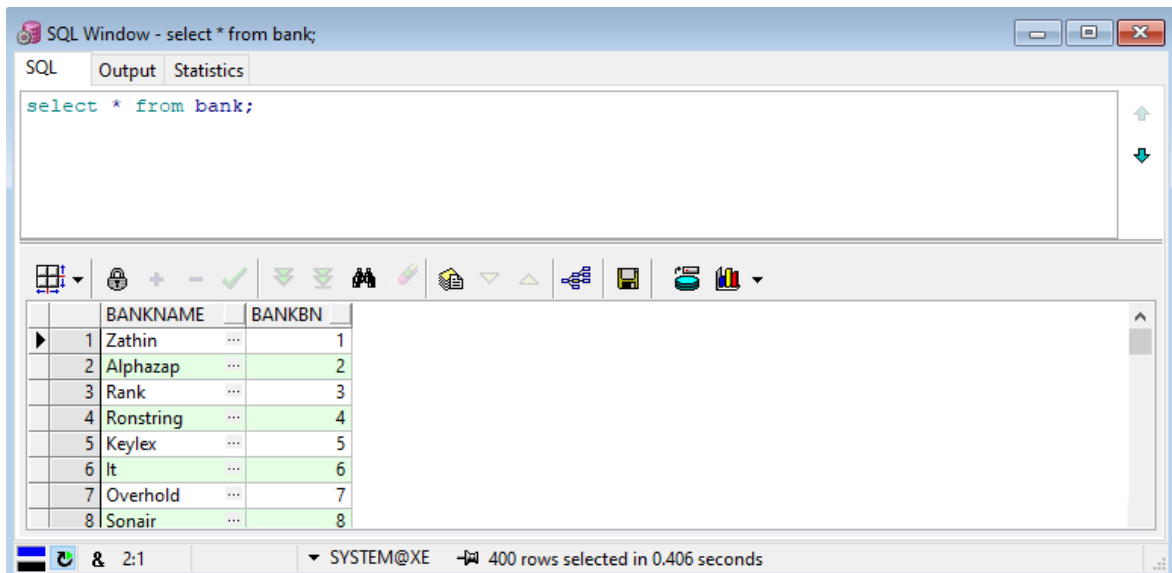
	BANKNAME	BANKBN
1	Zathin	1
2	Alphazap	2
3	Rank	3
4	Ronstring	4
5	Keylex	5
6	It	6
7	Overhold	7
8	Sonair	8

נמחק את 100 הרשומות האחרונות מטבלת Bank:

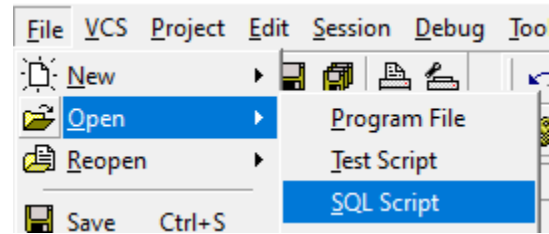


נציג את המידע לאחר המחיקה:

(ניתן לראות שנותרו 400 שורות מתוך 500 שהיו לפני המחיקה)

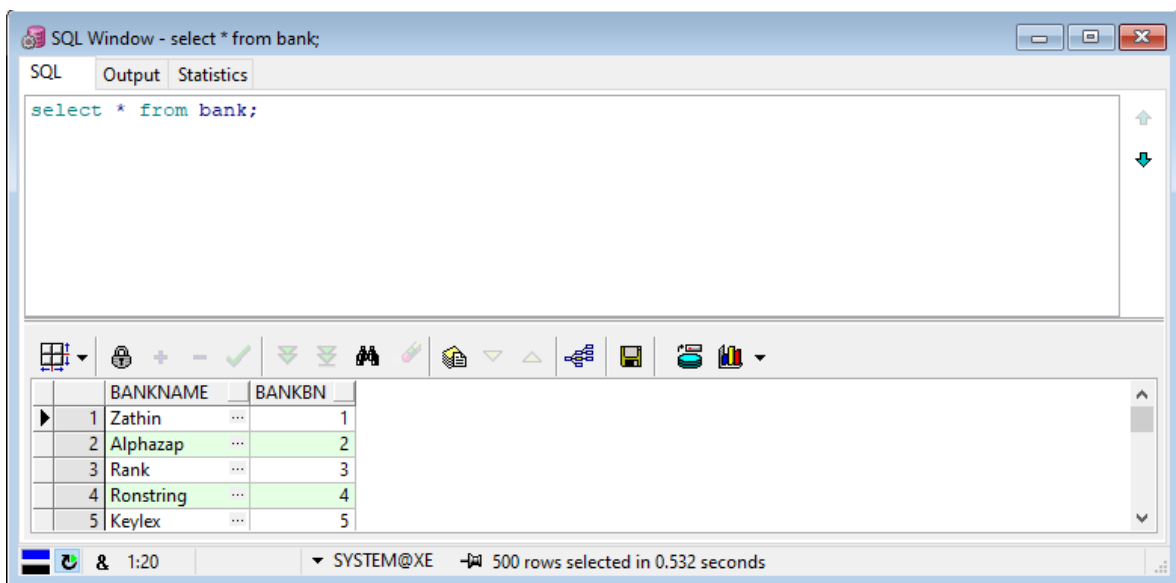


כעת נשתמש בגיבוי לצורך אחזור הרשומות שנמחקו:



נציג את הרשומות מחדש.

ניתן לראות כי כעת ישנן 500 רשומות והמידע אכן אוחד.



(1) פרטי המלווים שנתנו הלוואות בסכום כולל העולה על X (בדוגמא הנ"ל - 3000000)

SQL Output Statistics						
<pre> select * from lender where lenderbn in (select distinct lenderbn from lender natural join loan natural join payment group by lenderbn having sum(totalamount) > 3000000) </pre>						
	LENDERNAME	LENDERADDRESS	LENDERPHONE	LENDERMAIL	LENDERBN	
1	Misty Grigsby	57646 Raven Avenue	8703654200	mgrigsbyaw@rakuten.co.jp	393	
2	Edeline Scutcheon	62832 Arizona Pass	6956836082	escutcheona2@exblog.jp	363	
3	Cecil Shilton	2 Washington Park	9657426093	cshiltoncw@usgs.gov	465	
4	Sara-ann Lutsch	141 Crest Line Park	1461745183	slutsch7r@eepurl.com	280	
5	Selle Kettlestringes	9 Meadow Ridge Plaza	6683332431	skettlestringes4m@biblegateway.com	167	
6	Pattin Ormestone	4 Golf View Pass	5172659266	pormistone6u@wufoo.com	247	
7	Fifine Braun	3407 2nd Drive	7753049894	fbraunaj@newyorker.com	380	
8	Aigneis Seniour	4 Mayfield Alley	5404088833	aseniorcm@mozilla.com	455	
9	Benedicta Taylder	73597 Banding Plaza	7807874812	btaylder7g@blog.com	269	
10	Fleur Faulconer	41331 Stone Corner Plaza	5922745318	ffaulconer15@miitbeian.gov.cn	42	
11	Louise Rugge	559 Mallory Drive	1419306035	lrugge9u@addthis.com	355	
12	Jacquelin holmes	52 Melvin Place	7017916765	jholmesx@google.ca	34	
13	Micaela Corstorphine	158 Chinook Junction	3965855029	mcorstorphine8m@sciencedaily.com	311	
14	Maxwell Hathaway	0 Burning Wood Hill	9058031861	mhathawaydg@washingtonpost.com	485	
15	Birgitta Betteney	42 Melby Pass	3838495877	bbetteney48@lulu.com	153	
16	Carlos Davidowicz	22065 Harpers Center	6165208220	cdavidowicz4e@eventhite.com	172	

(2) הצגת קוד התשלומים בהם ללווה יש בעו"ש סכום גדול יותר מאשר סכום ההלוואה שלקח

SQL Output Statistics

```
select paycode
from payment natural join payedby natural join belongsto natural join bankaccount
where balance > totalamount;
```

	PAYCODE
1	132
2	134
3	490
4	144
5	565
6	328
7	486
8	627
9	228
10	407
11	591
12	464
13	191
14	518
15	387
16	591
17	599
18	284
19	176
20	518
21	191
22	112

SYSTEM@XE 261 rows selected in 0.437 seconds

(3) שמות ותז הלווים שלקחו יותר מ 1000 הלוואות

SQL			Output	Statistics
<pre>select pfullname, pid from person natural join loaner natural join payedby natural join payment natural join loan group by pid, pfullname having count(*) > 1000</pre>				
		PFULLNAME	PID	
▶	1	JimmieBates	10322	
	2	JudgeKier	11330	
	3	PhilSkarsgard	18153	
	4	BrendaKrumholtz	19628	
	5	ThomasQuinones	21467	
	6	ShirleyStamp	28300	
	7	JoyBelushi	20724	
	8	RosanneNivola	22499	
	9	BelindaMarsden	28727	
	10	XanderBarkin	29432	
	11	GiovanniWarden	18664	
	12	BrookeMcCormack	20139	
	13	JohnnieFox	21051	
	14	OzzyDalley	24322	
	15	LarryChaplin	17418	
	16	KennyDalley	17813	
	17	NikkaKeen	24045	
	18	KevnWithers	10585	
	19	MacHayek	16164	
1:22		SYSTEM@XE 58 rows selected in 0.578 seconds		

(4) ת.ז. + ממוצע ההלוואות של כל הנשיאים שחתמו על יותר מ-10 הלוואות וממוצע ההלוואות שעליהן חתמו גבוה מ-70,000

SQL		Output	Statistics
<pre>select distinct pidpr, avg(totalamount) from loan natural join payment group by pidpr having count(*) > 10 and avg(totalamount) > 70000</pre>			
	PIDPR	AVG(TOTALAMOUNT)	
1	17495	77328.0392156863	
2	17859	77206.9444444444	
3	15546	73529.7692307692	
4	29106	76548.9230769231	
5	27967	70937.0930232558	
6	15885	70133.0847457627	
7	29782	76366.4222222222	
8	19499	82368.025	
9	16043	77281.652173913	
10	13646	76368.8461538462	
11	24372	88431.8181818182	
12	12392	72728.5106382979	
13	11000	72374.2162162162	
14	10098	72252.7111111111	
15	28161	78016.6875	
16	29942	76734.7045454545	
17	14379	79781.2222222222	
18	19781	85604.0212765957	
19	25004	80640.4418604651	

SYSTEM@XE 330 rows selected in 0.172 seconds

(5) קוד (מק"ט) כל ההלוואות שהלווה שלהם מנהל חשבון בבנק מספר 1

SQL

Output

Statistics

```

select loanid
from loan natural join payment natural join payedby natural join belongsto
where bankbn = 1
group by loanid
    
```

	LOANID
1	11115
2	11116
3	11127
4	11129
5	11133
6	11138
7	11144
8	11147
9	11150
10	11175
11	11196
12	11197
13	11204
14	11215
15	11236
16	11255
17	11262
18	11270
19	11287

4:16
0:28
SYSTEM@XE
21410 rows selected in 28.281 seconds

(7) ת.ז הערבים שסכום ההלוואות עליהן הם חתומים גדול מסכום נתון (בדוגמא הנ"ל 1,000,000,000)

SQL	Output	Statistics
<pre>select pidgu, sum(totalamount) from person natural join guarantee natural join loan natural join payment group by pidgu having sum(totalamount) > 1000000000</pre>		
PIDGU	SUM(TOTALAMOUNT)	
1	21908	1489773033
2	21250	2124476127
3	14412	1321948674
4	20285	1733470092
5	15016	1355480406
6	11322	1342709271
7	21142	2208116673
8	15790	2130730623
9	28391	1544269023
10	16511	1657902627
11	22472	1642491081
12	23462	1212999786
13	17317	1625219910
14	14838	1684085121
15	23927	1707721596
16	27639	1423546785
17	19837	1578724155
18	15841	1889508276

4:32 0:09 SYSTEM@XE 506 rows selected in 9.579 seconds

(8) פרטי הערבים שחתומים על יותר מ X הלוואות (בדוגמא הנ"ל - 4)

SQL

Output

Statistics

```
select *
from person
where pid in (select distinct pidgu
from loan natural join guarantee natural join person
group by pidgu
having count (*) > 4)
```

	PFULLNAME	PID	PADDRESS	PPHONE	PMAIL
1	CollectiveMaguire	16407	87 Hughes Street	0565294591	collective.maguire@mastercardinternational
2	RandallMarx	29490	74 Webster Groves	0559010817	r.marx@progressivedesigns.com
3	JoyRussell	11942	9 Spears Ave	0520551637	joy.russell@diageo.fr
4	RhonaMcGoohan	16651	261 Berkley Blvd	0556668467	rhona.mcgoohan@privatebancorp.com
5	FranceDern	16511	60 Richmond Hill Street	0528955138	france.dern@staffforce.com
6	ColinClinton	11685	16 Vin Drive	0584251622	cclinton@mls.com
7	BelindaChristie	11720	22nd Street	0561790922	belindac@esteelauder.com
8	JaneanePepper	16996	79 Mifune Street	0580257274	jpepper@cynergydata.de
9	VerucaViterelli	15790	255 Shannon Road	0558447530	veruca.viterelli@eagleone.au
10	TerryShaye	15793	53 Ossie Street	0569813943	terry.shaye@stm.com
11	RupertGooding	15491	16 Redford Street	0546776740	rupertg@cardinalcartridge.br
12	ScottHedaya	15841	17 Rozenburg Blvd	0536095815	s.hedaya@campbellsoup.com
13	JoePurefoy	15728	87 Negbaur Road	0593250392	joe.p@multimedialive.jp
14	JuiceCoughlan	27639	41 Sheen Road	0539058545	juice.coughlan@randomwalk.com
15	BradleyAli	27785	6 Patty	0565911249	bradley.ali@gdi.pl
16	RoseFeuerstein	19857	56 Koyana Road	0555149898	rosef@hatworld.jp
17	JarvisDillon	19837	992 Slidel Road	0587454115	jarvis.dillon@infopros.pl
18	RascalShandling	28777	70 Nielsen Street	0535354460	rascal.shandling@campbellsoup.de
19	BobHauer	28657	62 Malina Road	0528940462	bob.hauer@meritsantechologies.ch

6:21

0:02

SYSTEM@XE

382 rows selected in 2.642 seconds

6:21 0:02 SYSTEM@XE 382 rows selected in 2.642 seconds

אינדקסינג:

An index is a performance-tuning method of allowing faster retrieval of records. An index creates an entry for each value that appears in the indexed columns.

טבלת השוואה לפני – אחרי האינדקסים:

בחנו אינדקסים שונים עבור שאילתות שונות וכן אינדקסים שונים עבור אותן שאילתות.

מס' שאילתה	זמן ללא אינדקס	זמן עם אינדקס	אינדקס חיובי / שלילי	אחוז השינוי
1	0.172	0.204	שלילי	18%+
4	0.172	0.453	שלילי	160%+
4	0.172	0.157	חיובי	9%-
5	28.281	6.702	חיובי	76%-
6	0.641	0.234	חיובי	63%-
7	9.579	1.859	חיובי	80%-

עבור שאילתה (4)

The screenshot shows two windows from SQL Developer. The top window displays the SQL command to create an index on the 'loan' table. The bottom window displays a query that selects distinct 'pidpr' and 'avg(totalamount)' from a natural join of 'loan' and 'payment' tables, filtered by a count and an average value. The query results are shown in a table with 9 rows.

SQL Window:

```
CREATE INDEX loan_president_index
on loan (pidpr);
```

Query Window:

```
select distinct pidpr, avg(totalamount)
from loan natural join payment
group by pidpr
having count(*) > 10 and avg(totalamount) > 70000
```

Query Results:

	PIDPR	AVG(TOTALAMOUNT)
1	17495	77328.0392156863
2	17859	77206.9444444444
3	15546	73529.7692307692
4	29106	76548.9230769231
5	27967	70937.0930232558
6	15885	70133.0847457627
7	29782	76366.4222222222
8	19499	82368.025
9	16043	77281.652173913

Status Bar: 330 rows selected in 0.157 seconds

עבור שאילתה (5)

```
SQL Output Statistics
CREATE INDEX belong_bankbn_index
on belongsto (bankbn)
```

SQL Output Statistics

```
select loanid
from loan natural join payment natural join payedby natural
where bankbn = 1
group by loanid
```

↑

↓

SQL Output Statistics

LOANID

1	11115
2	11116
3	11127
4	11129
5	11133
6	11138
7	11144
8	11147
9	11150
10	11175
11	11196
12	11197
13	11204
14	11215
15	11236

5:1 0:06 SYSTEM@XE 21410 rows selected in 6.702 second

עבור שאילתה (6)

SQL Output Statistics

```
CREATE INDEX payment_year_index
on payment (extract(year from startdate))
```

SQL Output Statistics

```
select loanid
from loan natural join payment
where extract(year from startdate) = 2020
group by loanid
```

	LOANID
1	11391
2	11396
3	13363
4	13485
5	13586
6	13790
7	14080
8	14109
9	14214
10	14410
11	14606
12	14792
13	15087
14	15234
15	15270
16	15696
17	15871
18	15946

SYSTEM@XE

556 rows selected in 0.234 seconds

אינדקסים לא-יעילים:

עבור שאילתה (1):

SQL Output Statistics

```
CREATE INDEX payment_totalamount_index
on payment (totalamount);
```

SQL Output Statistics

```
select *
from lender
where lenderbn in (select distinct lenderbn
from lender natural join loan natural join payment
group by lenderbn
having sum(totalamount) > 3000000)
```

	LENDERNAME	LENDERADDRESS	LENDERPHONE	LENDERMAIL	LENDERBN
1	Misty Grigsby	57646 Raven Avenue	8703654200	mgrigsbyaw@rakuten.co.jp	393
2	Edeline Scutcheon	62832 Arizona Pass	6956836082	escutcheona2@exblog.jp	363
3	Cecil Shilton	2 Washington Park	9657426093	cshiltoncw@usgs.gov	465
4	Sara-ann Lutsch	141 Crest Line Park	1461745183	slutsch7r@eepurl.com	280
5	Selle Kettlestringes	9 Meadow Ridge Plaza	6683332431	skettlestringes4m@biblegateway.com	167
6	Pattin Ormestone	4 Golf View Pass	5172659266	pormistone6u@wufoo.com	247
7	Fifine Braun	3407 2nd Drive	7753049894	fbraunaj@newyorker.com	380
8	Aigneis Senior	4 Mayfield Alley	5404088833	aseniorcm@mozilla.com	455
9	Benedicta Taylder	73597 Banding Plaza	7807874812	btaylder7g@blog.com	269
10	Fleur Faulconer	41331 Stone Corner Plaza	5922745318	ffaulconer15@miitbeian.gov.cn	42
11	Louise Rugge	559 Mallory Drive	1419306035	lrugge9u@addthis.com	355
12	Jacquelin holmes	52 Melvin Place	7017916765	jholmesx@google.ca	34
13	Micaela Corstorphine	158 Chinook Junction	3965855029	mcorstorphine8m@sciencedaily.com	311
14	Maxwell Hathaway	0 Burning Wood Hill	9058031861	mhathawaydg@washingtonpost.com	485
15	Birgitta Betteney	42 Melby Pass	3838495877	bbetteney48@lulu.com	153
16	Carlos Davidowicz	22065 Hancock Center	6165209220	cdavidowicz4r@earthlink.net	172

6:35

SYSTEM@XE 323 rows selected in 0.204 seconds

עבור שאילתה (4):

```
SQL Output Statistics
CREATE INDEX lender_lenderPhone_index
on lender (lenderPhone);
```

SQL Output Statistics

```
select distinct pidpr, avg(totalamount)
from loan natural join payment
group by pidpr
having count (*) > 10 and avg(totalamount) > 70000
```

	PIDPR	AVG(TOTALAMOUNT)
1	17495	77328.0392156863
2	17859	77206.9444444444
3	15546	73529.7692307692
4	29106	76548.9230769231
5	27967	70937.0930232558
6	15885	70133.0847457627
7	29782	76366.4222222222
8	19499	82368.025
9	16043	77281.652173913
10	13646	76368.8461538462
11	24372	88431.8181818182
12	12392	72728.5106382979
13	11000	72374.2162162162
14	10098	72252.7111111111
15	28161	78016.6875
16	29942	76734.7045454545
17	14379	79781.2222222222
18	19781	85604.0212765957
19	25904	80649.4418604651

SYSTEM@XE 330 rows selected in 0.453 seconds