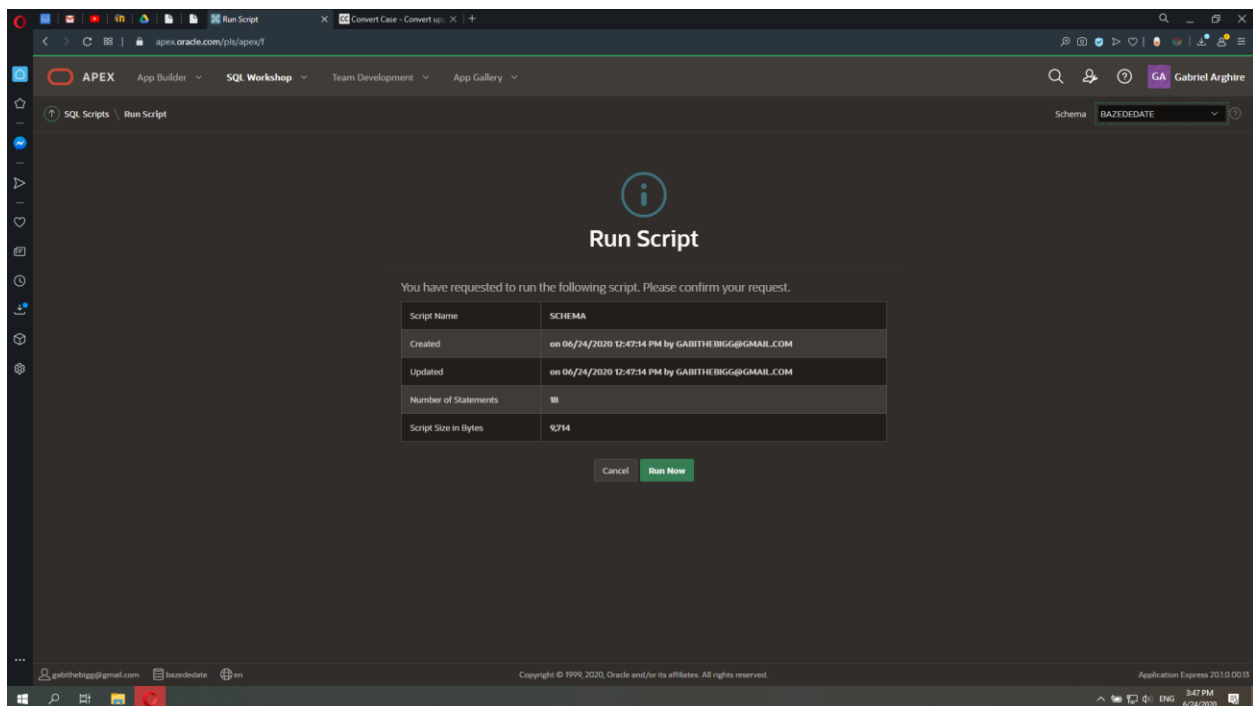


# Queries output

## Using Apex Oracle

---

Running script called "SCHEMA" for creation of tables, adding constraints and inserting data.



## Results from page 1, after running script "SCHEMA"

Script: SCHEMA Status: Complete View: Detail Summary Rows: 15

Number	Elapsed	Statement	Feedback	Rows
1	0.08	CREATE TABLE "EMPLOYEE_GAR" ("EMP_ID" INT NOT NULL, "ODD_	Table created.	0
2	0.06	CREATE TABLE "OFFICE_GAR" ("CODE" INT NOT NULL, "CITY" VA	Table created.	0
3	0.06	CREATE TABLE "CUSTOMER_GAR" ("CUST_ID" INT NOT NULL, "SAL	Table created.	0
4	0.04	CREATE TABLE "PAYMENT_GAR" ("BANK_NAME" VARCHAR(255) NOT	Table created.	0
5	0.04	CREATE TABLE "BANK_GAR" ("BANK_NAME" VARCHAR(20), "SWIFT	Table created.	0
6	0.06	CREATE TABLE "ORDER_GAR" ("ORDER_ID" INT NOT NULL, "CUST_	Table created.	0
7	0.05	ALTER TABLE "EMPLOYEE_GAR" ADD CONSTRAINT "EMPLOYEE_GAR_FK0"	Table altered.	0
8	0.04	ALTER TABLE "EMPLOYEE_GAR" ADD CONSTRAINT "EMPLOYEE_GAR_FK1"	Table altered.	0
9	0.04	ALTER TABLE "CUSTOMER_GAR" ADD CONSTRAINT "CUSTOMER_GAR_FK0"	Table altered.	0
10	0.05	ALTER TABLE "PAYMENT_GAR" ADD CONSTRAINT "PAYMENT_GAR_FK0" F	Table altered.	0
11	0.05	ALTER TABLE "PAYMENT_GAR" ADD CONSTRAINT "PAYMENT_GAR_FK1" F	Table altered.	0
12	0.05	ALTER TABLE "ORDER_GAR" ADD CONSTRAINT "ORDER_GAR_FK0" FOR	Table altered.	0
13	0.18	INSERT ALL INTO BANK_GAR (BANK_NAME, SWIFT_CODE, COMMISSIO	11 row(s) inserted.	11
14	0.05	INSERT ALL INTO OFFICE_GAR (CODE, CITY, PHONE, ADDRESS,	10 row(s) inserted.	10
15	0.05	INSERT ALL INTO EMPLOYEE_GAR (EMP_ID, ODD_WEEK_OFFICE_ID	10 row(s) inserted.	10

Download

row(s) 1 - 15 of 18

18 Statements Processed 18 Successful 0 With Errors

## Results from page 2, after running script "SCHEMA"

Script: SCHEMA Status: Complete View: Detail Summary Rows: 15

Number	Elapsed	Statement	Feedback	Rows
16	0.07	INSERT ALL INTO CUSTOMER_GAR (CUST_ID, SALES_EMP_ID, LAS	4 row(s) inserted.	4
17	0.04	INSERT ALL INTO PAYMENT_GAR (BANK_NAME, PAY_CUST_ID, PAY	7 row(s) inserted.	7
18	0.06	INSERT ALL INTO ORDER_GAR (ORDER_ID, CUST_ID, ORDER_DAT	5 row(s) inserted.	5

Download

Previous row(s) 16 - 18 of 18

18 Statements Processed 18 Successful 0 With Errors

gabrielhegga@gmail.com basedate en Copyright © 1999, 2020, Oracle and/or its affiliates. All rights reserved. Application Express 2010.0.0015 3:48 PM 6/24/2020

## Data from "BANK\_GAR" table

The screenshot shows the Oracle APEX SQL Workshop interface. On the left, the Object Browser lists several tables: BANK\_GAR, CUSTOMER\_GAR, EMPLOYEE\_GAR, OFFICE\_GAR, ORDER\_GAR, and PAYMENT\_GAR. The main panel displays the data for the BANK\_GAR table. The table has columns: EDIT, BANK\_NAME, SWIFT\_CODE, and COMMISSION. The data is as follows:

EDIT	BANK_NAME	SWIFT_CODE	COMMISSION
	BRD	BRDPROBU	1
	ALPHA BANK	ALBPROBU	2
	BANC POST S.A	BPOSPROBU	3
	BCR	RNCBPROBU	.05
	BT	BTBLPROBU	0
	IBR	BRMARPROBU	.25
	BANK OF CYPRUS	BCYPROBU	.35
	CEC BANK	CECEPROBU	1
	CTBANK	CTBPROBU	2
	BANK LEUMI	DAFBPROBU	15
	NO BANK	00000000	0

Download

row(s) 1 - 11 of 11

## Data from "CUSTOMER\_GAR" table

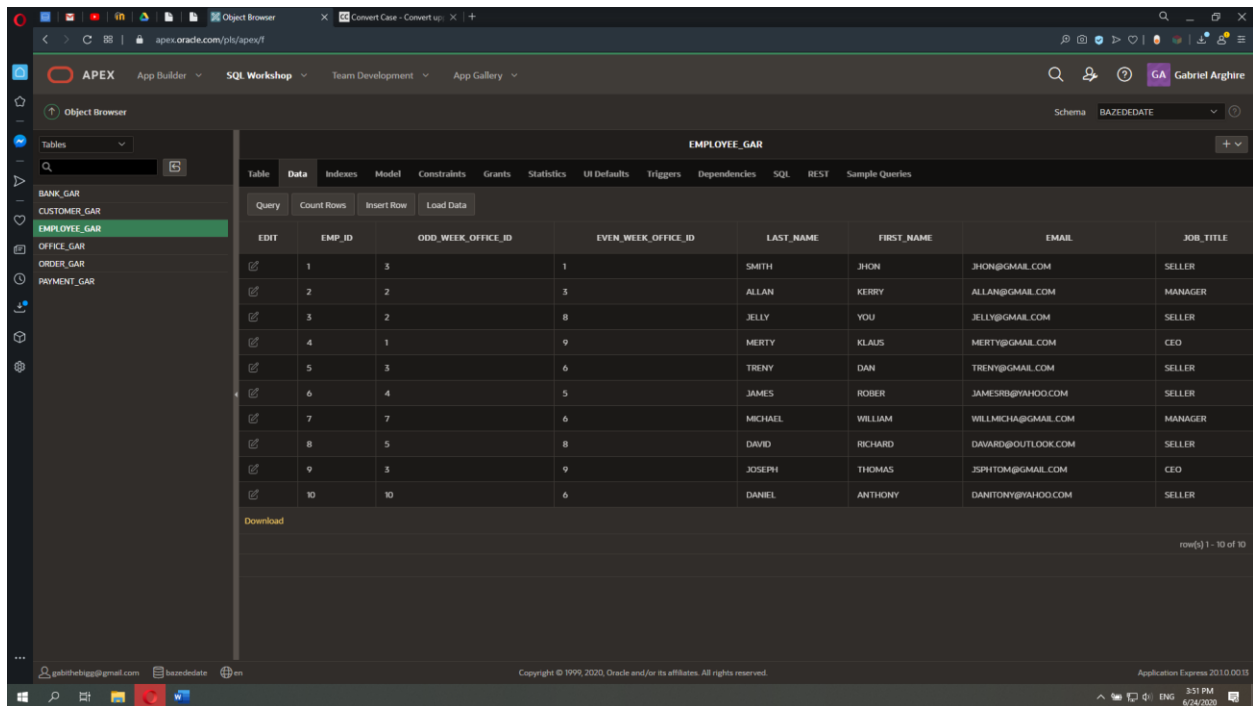
The screenshot shows the Oracle APEX SQL Workshop interface. On the left, the Object Browser lists several tables: BANK\_GAR, CUSTOMER\_GAR, EMPLOYEE\_GAR, OFFICE\_GAR, ORDER\_GAR, and PAYMENT\_GAR. The main panel displays the data for the CUSTOMER\_GAR table. The table has columns: EDIT, CUST\_ID, SALES\_EMP\_ID, LAST\_NAME, FIRST\_NAME, PHONE, ADDRESS, and EMAIL. The data is as follows:

EDIT	CUST_ID	SALES_EMP_ID	LAST_NAME	FIRST_NAME	PHONE	ADDRESS	EMAIL
	1	2	SMITH	KATE	14087859151	AVENUE OF THE ARMY	SMITHKATE@GMAIL.COM
	2	4	ALLAN	JHON	02078839232	DEVONSHIRE STREET	ALLANJH@GMAIL.COM
	3	1	MERELY	JOSH	14087859333	ST. JOHN STREET	MERJOSH@GMAIL.COM
	4	3	POPA	ALIN	40748599330	ORHIDEA STREET	ALINPOPA@GMAIL.COM

Download

row(s) 1 - 4 of 4

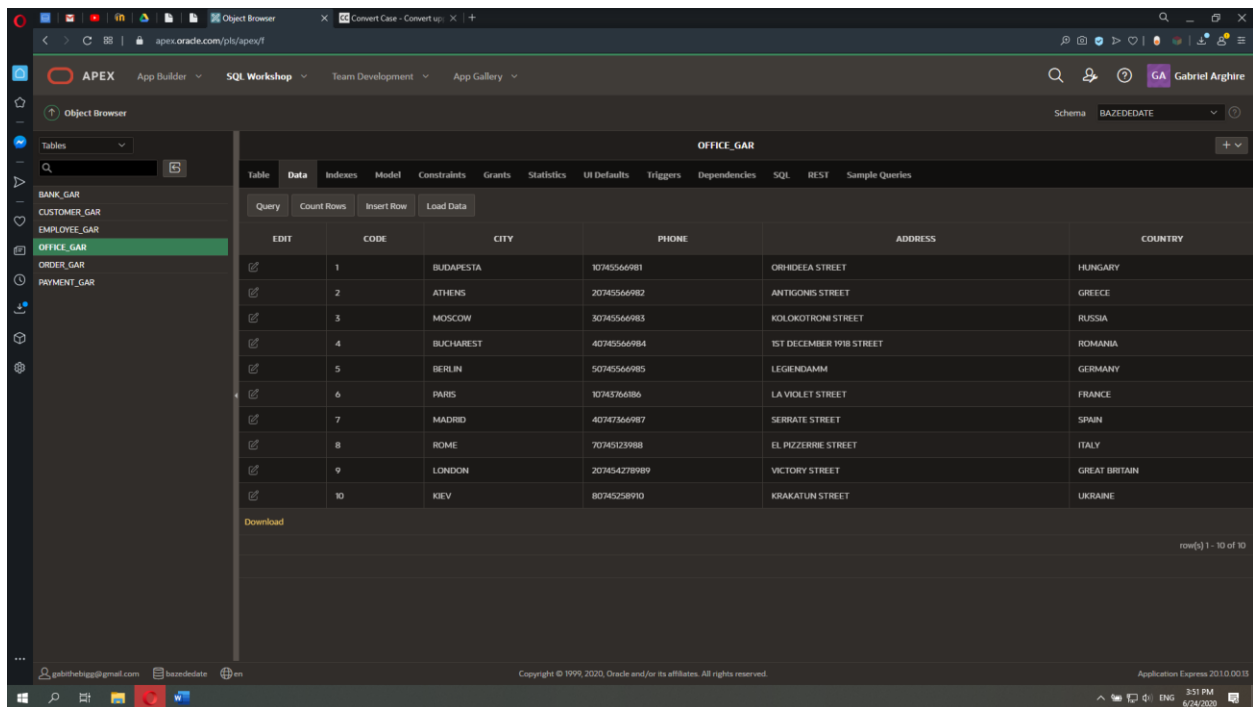
## Data from “EMPLOYEE\_GAR” table



The screenshot shows the Oracle APEX SQL Workshop interface. The left sidebar lists several tables: BANK\_GAR, CUSTOMER\_GAR, EMPLOYEE\_GAR (highlighted), ORDER\_GAR, and PAYMENT\_GAR. The main area displays the data for the EMPLOYEE\_GAR table. The table has columns: EMP\_ID, ODD\_WEEK\_OFFICE\_ID, EVEN\_WEEK\_OFFICE\_ID, LAST\_NAME, FIRST\_NAME, EMAIL, and JOB\_TITLE. There are 10 rows of data. The interface includes tabs for Query, Count Rows, Insert Row, and Load Data. The status bar at the bottom indicates 'row(s) 1 - 10 of 10'.

EMP_ID	ODD_WEEK_OFFICE_ID	EVEN_WEEK_OFFICE_ID	LAST_NAME	FIRST_NAME	EMAIL	JOB_TITLE
1	3	1	SMITH	JHON	JHON@GMAIL.COM	SELLER
2	2	3	ALLAN	KERRY	ALLAN@GMAIL.COM	MANAGER
3	2	8	JELLY	YOU	JELLY@GMAIL.COM	SELLER
4	1	9	MERTY	KLAUS	MERTY@GMAIL.COM	CEO
5	3	6	TRENY	DAN	TRENY@GMAIL.COM	SELLER
6	4	5	JAMES	ROBER	JAMESRB@YAHOO.COM	SELLER
7	7	6	MICHAEL	WILLIAM	WILLMICHA@GMAIL.COM	MANAGER
8	5	8	DAVID	RICHARD	DAVARD@OUTLOOK.COM	SELLER
9	3	9	JOSEPH	THOMAS	JSPH1TDM@GMAIL.COM	CEO
10	10	6	DANIEL	ANTHONY	DANTONY@YAHOO.COM	SELLER

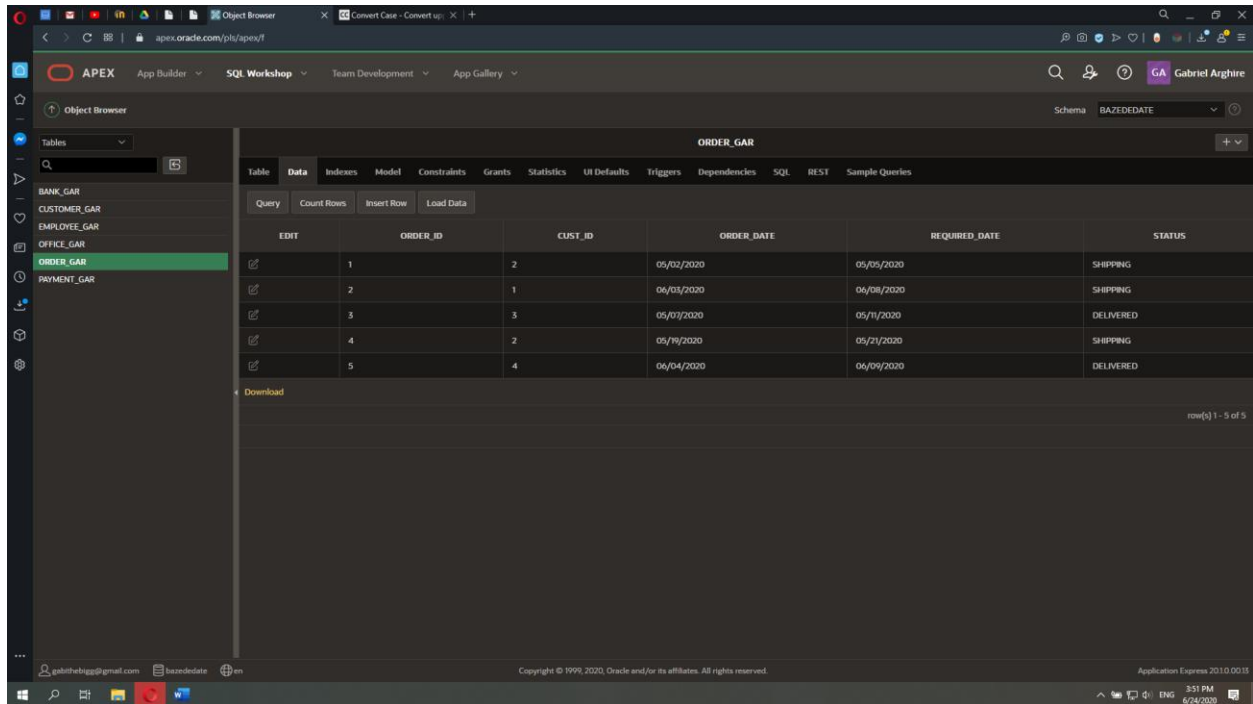
## Data from “OFFICE\_GAR” table



The screenshot shows the Oracle APEX SQL Workshop interface. The left sidebar lists several tables: BANK\_GAR, CUSTOMER\_GAR, EMPLOYEE\_GAR, OFFICE\_GAR (highlighted), ORDER\_GAR, and PAYMENT\_GAR. The main area displays the data for the OFFICE\_GAR table. The table has columns: CODE, CITY, PHONE, ADDRESS, and COUNTRY. There are 10 rows of data. The interface includes tabs for Query, Count Rows, Insert Row, and Load Data. The status bar at the bottom indicates 'row(s) 1 - 10 of 10'.

CODE	CITY	PHONE	ADDRESS	COUNTRY
1	BUDAPESTA	10745566981	ORHDEEA STREET	HUNGARY
2	ATHENS	20745566982	ANTIGONIS STREET	GREECE
3	MOSCOW	30745566983	KOLOKOTRONI STREET	RUSSIA
4	BUCHAREST	40745566984	1ST DECEMBER 1918 STREET	ROMANIA
5	BERLIN	50745566985	LEGENDAMM	GERMANY
6	PARIS	10745766986	LA VIOLET STREET	FRANCE
7	MADRID	40747566987	SERRATE STREET	SPAIN
8	ROME	70745323988	EL PIZZERRIE STREET	ITALY
9	LONDON	207454278989	VICTORY STREET	GREAT BRITAIN
10	KIEV	80745258990	KRAKATUN STREET	UKRAINE

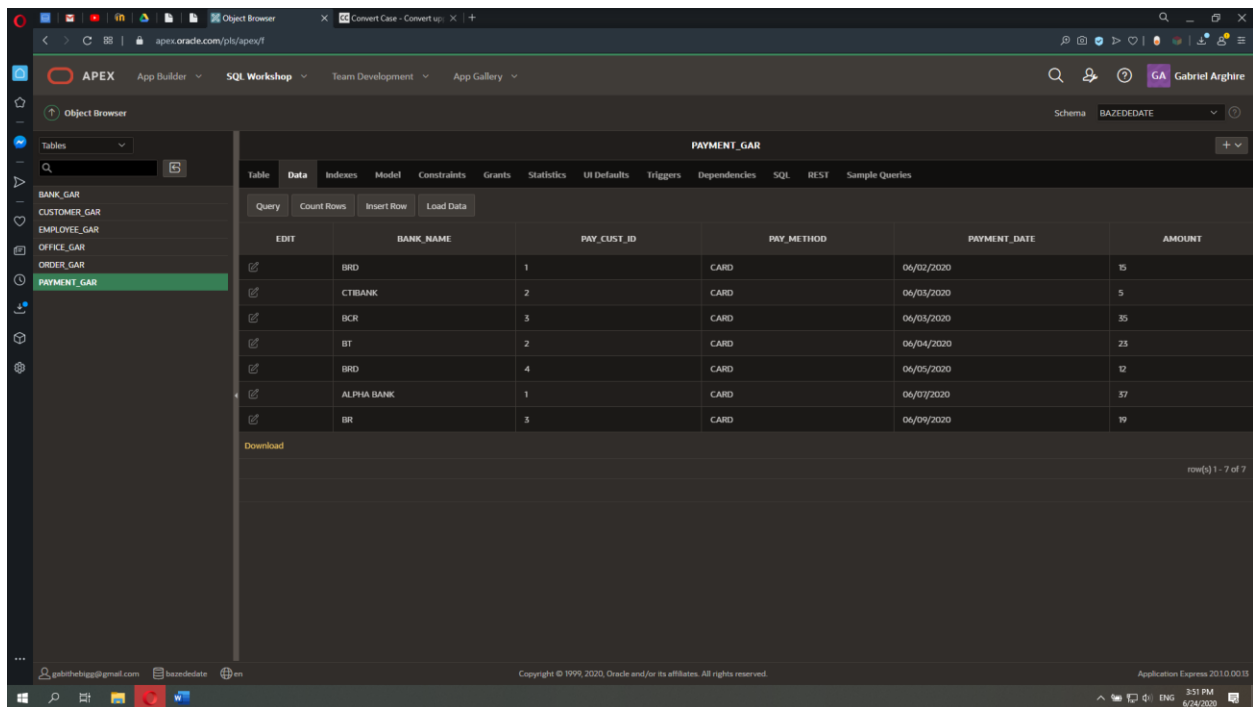
## Data from “ORDER\_GAR” table



The screenshot shows the Oracle APEX SQL Workshop interface. The left sidebar lists several tables: BANK\_GAR, CUSTOMER\_GAR, EMPLOYEE\_GAR, OFFICE\_GAR, ORDER\_GAR (highlighted), and PAYMENT\_GAR. The main area displays the data for the ORDER\_GAR table. The table has columns: ORDER\_ID, CUST\_ID, ORDER\_DATE, REQUIRED\_DATE, and STATUS. There are 5 rows of data.

ORDER_ID	CUST_ID	ORDER_DATE	REQUIRED_DATE	STATUS
1	2	05/02/2020	05/05/2020	SHIPPING
2	1	04/03/2020	04/08/2020	SHIPPING
3	3	05/03/2020	05/11/2020	DELIVERED
4	2	05/19/2020	05/21/2020	SHIPPING
5	4	04/04/2020	04/09/2020	DELIVERED

## Data from “PAYMENT\_GAR” table



The screenshot shows the Oracle APEX SQL Workshop interface. The left sidebar lists several tables: BANK\_GAR, CUSTOMER\_GAR, EMPLOYEE\_GAR, OFFICE\_GAR, ORDER\_GAR, and PAYMENT\_GAR (highlighted). The main area displays the data for the PAYMENT\_GAR table. The table has columns: BANK\_NAME, PAY\_CUST\_ID, PAY\_METHOD, PAYMENT\_DATE, and AMOUNT. There are 7 rows of data.

BANK_NAME	PAY_CUST_ID	PAY_METHOD	PAYMENT_DATE	AMOUNT
BRD	1	CARD	04/02/2020	15
CTEBANK	2	CARD	04/03/2020	5
BCR	3	CARD	04/03/2020	35
BT	2	CARD	04/04/2020	25
BRD	4	CARD	04/05/2020	12
ALPHA BANK	1	CARD	04/07/2020	37
BR	3	CARD	04/09/2020	19

Until here I have shown results of “SCHEMA” script.

Next pages are about displaying the results of the queries from the “SQL” script.

-- 1. GROUP BY, HAVING, COUNT

-- Count all the employees who are sellers.

The screenshot shows the Oracle APEX SQL Workshop interface. The SQL command window contains the following query:

```
1 -- 1. GROUP BY, HAVING  
2 -- Count all the employees who are sellers.  
3  
4 SELECT COUNT(EMP_ID), JOB_TITLE  
5 FROM EMPLOYEE_GAR  
6 GROUP BY JOB_TITLE  
7 HAVING JOB_TITLE LIKE 'SELLER';  
8  
9  
10 -- 2. GROUP BY, HAVING, ORDER BY  
11  
12 -- Display, in descending order,  
13 -- all the offices, grouped by the country they are in.  
14 -- (Only include offices in east european countries)  
15  
16 SELECT COUNT(CODE), COUNTRY  
17 FROM OFFICE_GAR  
18 GROUP BY COUNTRY  
19 HAVING COUNTRY IN ('ROMANIA', 'RUSSIA', 'UKRAINE')  
20 ORDER BY COUNT(CODE) DESC;  
21  
22 -- 3. SUBSTR, WHERE  
23  
24 -- Display all the orders that were requested in 2020 and also delivered.  
25  
26 SELECT *  
27 FROM ORDER_GAR
```

The Results tab shows the output of the first query:

COUNT(EMP_ID)	JOB_TITLE
6	SELLER

1 rows returned in 0.01 seconds

-- 2. GROUP BY, HAVING, ORDER BY

-- Display, in descending order,

-- all the offices, grouped by the country they are in.

-- (Only include offices in east european countries)

The screenshot shows the Oracle APEX SQL Workshop interface. The SQL command window contains the same query as the previous screenshot:

```
1 -- 1. GROUP BY, HAVING  
2 -- Count all the employees who are sellers.  
3  
4 SELECT COUNT(EMP_ID), JOB_TITLE  
5 FROM EMPLOYEE_GAR  
6 GROUP BY JOB_TITLE  
7 HAVING JOB_TITLE LIKE 'SELLER';  
8  
9  
10 -- 2. GROUP BY, HAVING, ORDER BY  
11  
12 -- Display, in descending order,  
13 -- all the offices, grouped by the country they are in.  
14 -- (Only include offices in east european countries)  
15  
16 SELECT COUNT(CODE), COUNTRY  
17 FROM OFFICE_GAR  
18 GROUP BY COUNTRY  
19 HAVING COUNTRY IN ('ROMANIA', 'RUSSIA', 'UKRAINE')  
20 ORDER BY COUNT(CODE) DESC;  
21  
22 -- 3. SUBSTR, WHERE  
23  
24 -- Display all the orders that were requested in 2020 and also delivered.  
25  
26 SELECT *  
27 FROM ORDER_GAR
```

The Results tab shows the output of the second query:

COUNT(CODE)	COUNTRY
1	RUSSIA
1	UKRAINE
1	ROMANIA

3 rows returned in 0.01 seconds

### -- 3. SUBSTR, WHERE

-- Display all the orders that were requested in 2020 and also delivered.

The screenshot shows the SQL Workshop interface with a query editor and a results table. The query is as follows:

```
-- Display, in ascending order,
-- all the offices, grouped by the country they are in.
-- (Only include offices in east european countries)
13
14
15
16 SELECT COUNT(CODE), COUNTRY
17 FROM OFFICE_GAR
18 GROUP BY COUNTRY
19 HAVING COUNTRY IN ('ROMANIA', 'RUSSIA', 'UKRAINE')
20 ORDER BY COUNT(CODE) DESC;
21
22 -- 3. SUBSTR, WHERE
23
24 -- Display all the orders that were requested in 2020 and also delivered.
25
26 SELECT *
27 FROM ORDER_GAR
28 WHERE SUBSTR(ORDER_DATE, 7, 4) = '2020'
29 AND STATUS = 'DELIVERED';
30
31 -- 4. DECODE
32
33 -- Check which bank SWIFT code is BRDEROBU.
34
35 SELECT BANK_NAME,
36 DECODE (SWIFT_CODE, 'BRDEROBU', SWIFT_CODE,
37 'NO RESULT') RESULT
38 FROM BANK_GAR;
39
```

The results table shows 2 rows returned in 0.01 seconds:

ORDER_ID	CUST_ID	ORDER_DATE	REQUIRED_DATE	STATUS
5	3	05/07/2020	05/11/2020	DELIVERED
5	4	06/04/2020	06/09/2020	DELIVERED

### -- 4. DECODE

-- Check which bank SWIFT code is BRDEROBU.

The screenshot shows the SQL Workshop interface with a query editor and a results table. The query is as follows:

```
-- 4. DECODE
-- Check which bank SWIFT code is BRDEROBU.
30
31
32
33
34
35 SELECT BANK_NAME,
36 DECODE (SWIFT_CODE, 'BRDEROBU', SWIFT_CODE,
37 'NO RESULT') RESULT
38 FROM BANK_GAR;
39
40 -- 5. CASE, NULL
41
42 -- Order customers by their address.
```

The results table shows 11 rows returned in 0.00 seconds:

BANK_NAME	RESULT
BRD	BRDEROBU
ALPHA BANK	NO RESULT
BANK POST S.A.	NO RESULT
BCR	NO RESULT
BT	NO RESULT
BR	NO RESULT
BANK OF CYPRUS	NO RESULT
CEC BANK	NO RESULT
CTIBANK	NO RESULT
BANK LEUMI	NO RESULT
NO BANK	NO RESULT

-- 5. CASE, NULL

-- Order customers by their address.

-- If address is null, order by their last name

The screenshot shows the SQL Workshop interface with a query that uses a CASE statement to handle NULL values in the ORDER BY clause. The query selects customer information and orders it by address, with a fallback to last name if the address is NULL. The results table shows 4 rows of customer data.

```
36 DECODE (SHIFT_CODE, 'BRODERBU', SHIFT_CODE,
37        'NO RESULT') RESULT
38 FROM BANK_GAR;
39
40 -- 5. CASE, NULL
41
42 -- Order customers by their address.
43 -- If address is null, order by their last name
44
45 SELECT *
46 FROM CUSTOMER_GAR
47 ORDER BY
48 (CASE
49  WHEN ADDRESS IS NULL THEN LAST_NAME
50  ELSE ADDRESS
51 END);
52
53 -- 6. INNER JOIN
54
55 -- Display which order belongs to which customer,
56 -- ordered by most recent orders.
57
58 SELECT *
59 FROM ORDER_GAR OG
60 INNER JOIN CUSTOMER_GAR CG ON OG.CUST_ID = CG.CUST_ID
61 ORDER BY OG.ORDER_DATE DESC;
```

CUST_ID	SALES_EMP_ID	LAST_NAME	FIRST_NAME	PHONE	ADDRESS	EMAIL
1	2	SMITH	KATE	14087859151	AVENUE OF THE ARMY	SMITHKATE@GMAIL.COM
2	4	ALLAN	JHON	02071839232	DEVONSHIRE STREET	ALLANJH@GMAIL.COM
4	3	POPA	ALIN	40748599330	ORHIDEEA STREET	ALINPOPA@GMAIL.COM
5	1	MERELY	JOSH	14087859333	ST. JOHN STREET	MERJOSH@GMAIL.COM

-- 6. INNER JOIN

-- Display which order belongs to which customer,

-- ordered by most recent orders.

The screenshot shows the SQL Workshop interface with a query that uses an INNER JOIN to link orders to customers and orders the results by order date in descending order. The results table shows 5 rows of order data.

```
48 (CASE
49  WHEN ADDRESS IS NULL THEN LAST_NAME
50  ELSE ADDRESS
51 END);
52
53 -- 6. INNER JOIN
54
55 -- Display which order belongs to which customer,
56 -- ordered by most recent orders.
57
58 SELECT *
59 FROM ORDER_GAR OG
60 INNER JOIN CUSTOMER_GAR CG ON OG.CUST_ID = CG.CUST_ID
61 ORDER BY OG.ORDER_DATE DESC;
62
63 -- 7. RIGHT JOIN
64
65 -- Display all the employees that had or not customers
66
67 SELECT *
68 FROM CUSTOMER_GAR CG
69 RIGHT JOIN EMPLOYEE_GAR EG ON CG.SALES_EMP_ID = EG.EMP_ID;
```

ORDER_ID	CUST_ID	ORDER_DATE	REQUIRED_DATE	STATUS	CUST_ID	SALES_EMP_ID	LAST_NAME	FIRST_NAME	PHONE	ADDRESS	EMAIL
5	4	06/04/2020	06/09/2020	DELIVERED	4	3	POPA	ALIN	40748599330	ORHIDEEA STREET	ALINPOPA@GMAIL.COM
2	1	06/03/2020	06/08/2020	SHIPPING	1	2	SMITH	KATE	14087859151	AVENUE OF THE ARMY	SMITHKATE@GMAIL.COM
4	2	05/19/2020	05/21/2020	SHIPPING	2	4	ALLAN	JHON	02071839232	DEVONSHIRE STREET	ALLANJH@GMAIL.COM
3	3	05/07/2020	05/11/2020	DELIVERED	3	1	MERELY	JOSH	14087859333	ST. JOHN STREET	MERJOSH@GMAIL.COM
1	2	05/02/2020	05/05/2020	SHIPPING	2	4	ALLAN	JHON	02071839232	DEVONSHIRE STREET	ALLANJH@GMAIL.COM



## -- 7. RIGHT JOIN

-- Display all the employees that had or not customers

The screenshot shows the Oracle APEX SQL Workshop interface. The SQL command window contains the following query:

```
-- 7. RIGHT JOIN
-- Display all the employees that had or not customers
SELECT *
FROM CUSTOMER_GAR CG
RIGHT JOIN EMPLOYEE_GAR EG ON CG.SALES_EMP_ID = EG.EMP_ID;
-- 8. UNION
```

The Results tab displays 10 rows. The columns are: CUST\_ID, SALES\_EMP\_ID, LAST\_NAME, FIRST\_NAME, PHONE, ADDRESS, EMAIL, EMP\_ID, ODD\_WEEK\_OFFICE\_ID, EVEN\_WEEK\_OFFICE\_ID, LAST\_NAME, FIRST\_NAME, EMAIL, and JOB\_ID. The data is as follows:

CUST_ID	SALES_EMP_ID	LAST_NAME	FIRST_NAME	PHONE	ADDRESS	EMAIL	EMP_ID	ODD_WEEK_OFFICE_ID	EVEN_WEEK_OFFICE_ID	LAST_NAME	FIRST_NAME	EMAIL	JOB_ID
1	2	SMITH	KATE	14087859131	AVENUE OF THE ARMY	SMITHKATE@GMAIL.COM	2	2	3	ALLAN	KERRY	ALLAN@GMAIL.COM	MANA
2	4	ALLAN	JHON	0207839232	DEVONSHIRE STREET	ALLANJH@GMAIL.COM	4	1	9	MERTY	KLAUS	MERTY@GMAIL.COM	CEO
3	1	MERELY	JOSH	14087859333	ST. JOHN STREET	MERJOSH@GMAIL.COM	1	3	1	SMITH	JHON	JHON@GMAIL.COM	SELLE
4	3	POPA	ALIN	40748599330	ORHIDEEA STREET	ALINPOPA@GMAIL.COM	3	2	8	JELLY	YOU	JELLY@GMAIL.COM	SELLE
-	-	-	-	-	-	-	6	4	5	JAMES	ROBER	JAMESRB@VAHOO.COM	SELLE
-	-	-	-	-	-	-	7	7	6	MICHAEL	WILLIAM	WILLMICHA@GMAIL.COM	MANA
-	-	-	-	-	-	-	8	5	8	DAVID	RICHARD	DAWARD@OUTLOOK.COM	SELLE
-	-	-	-	-	-	-	5	3	6	TRENY	DAN	TRENY@GMAIL.COM	SELLE
-	-	-	-	-	-	-	10	10	6	DANIEL	ANTHONY	DANTONY@VAHOO.COM	SELLE
-	-	-	-	-	-	-	9	3	9	JOSEPH	THOMAS	JSPHITOM@GMAIL.COM	CEO

## -- 8. UNION

-- Display all the distinct last names from both customers and employee tables

-- ordered in ascending order by their last name

The screenshot shows the Oracle APEX SQL Workshop interface. The SQL command window contains the following query:

```
-- 8. UNION
-- Display all the distinct last names from both customers and employee tables
-- ordered in ascending order by their last name
SELECT LAST_NAME FROM CUSTOMER_GAR
UNION
SELECT LAST_NAME FROM EMPLOYEE_GAR
ORDER BY LAST_NAME;
```

The Results tab displays 12 rows of distinct last names, ordered alphabetically:

LAST_NAME
ALLAN
DANIEL
DAVID
JAMES
JELLY
JOSEPH
MERELY
MERTY
MICHAEL
POPA
SMITH
TRENY

## -- 9. INTERSECT

-- Display all the identical last names from both customers and employee tables

The screenshot shows the Oracle SQL Workshop interface. The SQL Commands window contains the following code:

```
-- 9. INTERSECT
-- Display all the identical last names from both customers and employee tables
76 SELECT LAST_NAME FROM CUSTOMER_GAR
77 UNION
78 SELECT LAST_NAME FROM EMPLOYEE_GAR
79 ORDER BY LAST_NAME;
80
81 -- 9. INTERSECT
82
83 -- Display all the identical last names from both customers and employee tables
84
85 SELECT LAST_NAME FROM CUSTOMER_GAR
86 INTERSECT
87 SELECT LAST_NAME FROM EMPLOYEE_GAR;
88
89 -- 10. AVG
90
91 -- 11. COUNT
92
93 -- 12. MIN, MAX
94
95 -- 13. SELECT, FROM, WHERE, HAVING SUBQUERY
96
97 -- 14. SELECT, FROM, WHERE, HAVING SUBQUERY
98
99 -- 15. DIVISION OPERATOR
100
```

The Results window shows the output of the INTERSECT query:

LAST_NAME
ALLAN
SMITH

2 rows returned in 0.01 seconds

## -- 10. AVG, SUBQUERY, WHERE

-- Display all the payments, where the paid amount is greater

-- than the average paid amount, ordered by amount.

The screenshot shows the Oracle SQL Workshop interface. The SQL Commands window contains the following code:

```
-- 10. AVG, SUBQUERY, WHERE
-- Display all the payments, where the paid amount is greater
-- than the average paid amount, ordered by amount.
94 SELECT *
95 FROM PAYMENT_GAR
96 WHERE AMOUNT > (
97 SELECT AVG(AMOUNT)
98 FROM PAYMENT_GAR
99 )
100 ORDER BY AMOUNT DESC;
101
102 -- 11. COUNT, GROUP BY, HAVING
103
```

The Results window shows the output of the query:

BANK_NAME	PAY_CUST_ID	PAY_METHOD	PAYMENT_DATE	AMOUNT
ALPHA BANK	1	CARD	06/07/2020	37
BCR	3	CARD	06/03/2020	35
BT	2	CARD	06/04/2020	23

3 rows returned in 0.02 seconds

## -- 11. COUNT, GROUP BY, HAVING

-- Count all the employees whose last name start with letter A,

-- grouped by their last name.

The screenshot shows the SQL Workshop interface with the following SQL commands:

```
-- 10. AVG, SUBQUERY, WHERE
-- Display all the payments, where the paid amount is greater
-- than the average paid amount, ordered by amount.
SELECT *
FROM PAYMENT_GAR
WHERE AMOUNT > (
  SELECT AVG(AMOUNT)
  FROM PAYMENT_GAR
)
ORDER BY AMOUNT DESC;

-- 11. COUNT, GROUP BY, HAVING
-- Count all the employees whose last name start with letter A,
-- grouped by their last name.
SELECT COUNT(EMP_ID), LAST_NAME
FROM EMPLOYEE_GAR
GROUP BY LAST_NAME
HAVING LAST_NAME LIKE 'A%';

-- 12. MIN, MAX, UNION ALL
-- Display the minimum paid amount in the history of the shop
-- and the maximum, also, even it is the same value.
```

The results table shows the following data:

COUNT(EMP_ID)	LAST_NAME
1	ALLAN

1 rows returned in 0.00 seconds

## -- 12. MIN, MAX, UNION ALL

-- Display the minimum paid amount in the history of the shop

-- and the maximum, also, even it is the same value.

The screenshot shows the SQL Workshop interface with the following SQL commands:

```
-- 11. COUNT, GROUP BY, HAVING
-- Count all the employees whose last name start with letter A,
-- grouped by their last name.
SELECT COUNT(EMP_ID), LAST_NAME
FROM EMPLOYEE_GAR
GROUP BY LAST_NAME
HAVING LAST_NAME LIKE 'A%';

-- 12. MIN, MAX, UNION ALL
-- Display the minimum paid amount in the history of the shop
-- and the maximum, also, even it is the same value.
SELECT MIN(AMOUNT)
FROM PAYMENT_GAR
UNION ALL
SELECT MAX(AMOUNT)
FROM PAYMENT_GAR;

-- 13. SELECT, FROM, WHERE, IN SUBQUERY
-- Display all employees who have the same last name
-- as the customers.
```

The results table shows the following data:

MIN(AMOUNT)
5
37

2 rows returned in 0.00 seconds

-- 13. SELECT, FROM, WHERE, IN SUBQUERY

-- Display all employees who have the same last name

-- as the customers.

The screenshot shows the SQL Workshop interface with a query that uses a subquery to find employees whose last names match those of customers. The query is as follows:

```
-- 12. MIN, MAX, UNION ALL
-- Display the minimum paid amount in the history of the shop
-- and the maximum, also, even it is the same value.
SELECT MIN(AMOUNT)
FROM PAYMENT_GAR
UNION ALL
SELECT MAX(AMOUNT)
FROM PAYMENT_GAR;

-- 13. SELECT, FROM, WHERE, IN SUBQUERY
-- Display all employees who have the same last name
-- as the customers.
SELECT *
FROM EMPLOYEE_GAR
WHERE LAST_NAME IN (
    SELECT LAST_NAME
    FROM CUSTOMER_GAR
);
```

The results table shows two rows:

EMP_ID	ODD_WEEK_OFFICE_ID	EVEN_WEEK_OFFICE_ID	LAST_NAME	FIRST_NAME	EMAIL	JOB_TITLE
1	3	1	SMITH	JHON	JHON@GMAIL.COM	SELLER
2	2	3	ALLAN	KERRY	ALLAN@GMAIL.COM	MANAGER

-- 14. SELECT, FROM, WHERE, IN SUBQUERY

-- Display all the banks who are used in payments by customers.

The screenshot shows the SQL Workshop interface with a query that uses a subquery to find banks used in payments by customers. The query is as follows:

```
-- 14. SELECT, FROM, WHERE, IN SUBQUERY
-- Display all the banks who are used in payments by customers.
SELECT *
FROM PAYMENT_GAR
WHERE BANK_NAME IN (
    SELECT BANK_NAME
    FROM BANK_GAR
);
```

The results table shows seven rows:

BANK_NAME	PAY_CUST_ID	PAY_METHOD	PAYMENT_DATE	AMOUNT
BRD	1	CARD	06/02/2020	15
CTIBANK	2	CARD	06/03/2020	5
BCR	3	CARD	06/03/2020	35
BT	2	CARD	06/04/2020	23
BRD	4	CARD	06/05/2020	12
ALPHA BANK	1	CARD	06/07/2020	37
BR	3	CARD	06/09/2020	19

-- 15. DIVISION (NOT EXISTS) OPERATOR (with subquery)

-- Display all the employers that do not have yet customers

The screenshot shows the Oracle APEX SQL Workshop interface. The SQL Commands pane contains the following code:

```
136 -- Display all the banks who are used in payments by customers.
137
138 SELECT *
139 FROM PAYMENT_GAR
140 WHERE BANK_NAME IN (
141     SELECT BANK_NAME
142     FROM BANK_GAR
143 );
144
145 -- 15. DIVISION (NOT EXISTS) OPERATOR (with subquery)
146
147 -- Display all the employers that do not have yet customers
148
149 SELECT EG.EMP_ID, EG.LAST_NAME, EG.FIRST_NAME
150 FROM EMPLOYEE_GAR EG
151 WHERE NOT EXISTS (
152     SELECT *
153     FROM CUSTOMER_GAR CG
154     WHERE EG.EMP_ID = CG.SALES_EMP_ID
155 );
156
```

The Results pane shows the output of the query, displaying 6 rows of employee data:

EMP_ID	LAST_NAME	FIRST_NAME
6	JAMES	ROBER
7	MICHAEL	WILLIAM
8	DAVID	RICHARD
5	TRENY	DAN
10	DANIEL	ANTHONY
9	JOSEPH	THOMAS

6 rows returned in 0.01 seconds