

- Queries

1. Information about all the couriers sent more than 500km away.

SQL Code:

```
SELECT * FROM COURIER AS C JOIN RATE AS R ON
(C.RATE_ID=R.RATE_ID) WHERE R.C_DISTANCE>500;
```

Relational Algebra:

$$\pi_{(c.c_id)}(\rho(r, \sigma_{<r.c_distance>500>}rate) \bowtie_{<c.rate_id=d.rate_id>} \rho(c, courier))$$

Output:

The screenshot shows a PostgreSQL query editor with the following query:

```
1 SET search_path TO TejCourierService;
2
3 SELECT * FROM COURIER AS C JOIN RATE AS R ON (C.RATE_ID=R.RATE_ID) WHERE R.C_DISTANCE>500;
```

The output table is as follows:

c_id	c_name	c_dispatch_date	c_status	c_delivery_date	s_id	rate_id	p_id
1 C002	DEF	2020-08-08	DISPATCHED	2020-08-15	S013	RN101000	P111
2 C003	GHI	2020-08-05	DISPATCHED	2020-08-12	S011	RF102000	P112
3 C004	JKL	2020-08-10	PACKED	2020-08-17	S010	RF502000	P113
4 C006	MNO	2020-09-02	DELIVERED	2020-09-10	S014	RF11000	P115
5 C007	MNO	2020-05-02	FAILED	2020-05-10	S014	RF11000	P115

The screenshot shows a PostgreSQL query editor with the following query:

```
1 SET search_path TO TejCourierService;
2
3 SELECT * FROM COURIER AS C JOIN RATE AS R ON (C.RATE_ID=R.RATE_ID) WHERE R.C_DISTANCE>500;
```

The output table is as follows:

b_id	t_id	e_id	rate_id	c_weight	c_distance	c_type	c_rate
B101	T10102	E112	RN101000	10	1000	NON-FRAGILE	140
B101	T10103	E113	RF102000	10	2000	FRAGILE	600
B102	T10201	E112	RF502000	50	2000	FRAGILE	1000
B103	T10301	E132	RF11000	1	1000	FRAGILE	200
B103	T10301	E132	RF11000	1	1000	FRAGILE	200

2. Name of the branch which sent the maximum number of couriers in month of August.

SQL Code:

```
SELECT B_NAME FROM COURIER JOIN BRANCH ON
COURIER.B_ID=BRANCH.B_ID WHERE C_DISPATCH_DATE=(SELECT
MAX(C_DISPATCH_DATE) FROM COURIER WHERE C_DISPATCH_DATE
BETWEEN '#2020-08-01#' AND '#2020-08-19#');
```

Relational Algebra:

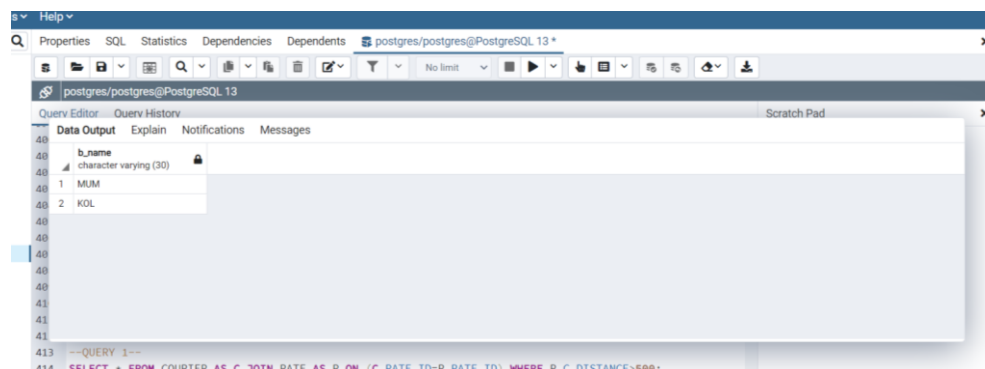
$$r1 \leftarrow \sigma_{\langle c_dispatch_date > 2020-08-01 \text{ AND } c_dispatch_date < 2020-08-31 \rangle} (\text{courier})$$

$$r2 \leftarrow \mathcal{F}_{\max \langle c_dispatch_date \rangle} (r1)$$

$$r3 \leftarrow \sigma_{\langle c_dispatch_date = r2 \rangle} \text{Courier} \bowtie_{\langle \text{courier.B_ID} = \text{Branch.B_ID} \rangle} \text{Branch}$$

$$\text{result} \leftarrow \Pi_{B_Name} (r3)$$

Output:



3. The average cost of courier in September.

SQL Code:

```
SELECT AVG(C_RATE) FROM RATE AS R JOIN COURIER AS C ON
(R.RATE_ID = C.RATE_ID) WHERE C.C_STATUS='DELIVERED' AND
C_DELIVERY_DATE BETWEEN '#2020-09-01#' AND '#2020-09-30#';
```

Relational Algebra:

$$r1 \leftarrow \rho(c, \sigma_{\langle c_delivery_date > 2020-09-01 \text{ AND } \langle c_delivery_date < 2020-09-31 \text{ AND } c_status = 'delivered' \rangle} \text{ courier})$$

$$r2 \leftarrow \rho(r, \text{rate})$$

result $\leftarrow \mathcal{F}_{\text{AVG}(C_RATE)} r1 \bowtie_{<c.rate_id=r.rate_id>} r2$

Output:

The screenshot shows a PostgreSQL Query Editor window. The query is as follows:

```

1
2
3 WHERE R.C_DISTANCE>500;
4
5 = C.RATE_ID) WHERE C.C_STATUS='DELIVERED' AND C.DELIVERY_DATE BETWEEN '2020-09-01' AND '2020-09-30';

```

The Data Output tab shows the following result:

avg
200.0000000000000000

4. To retrieve the status of couriers that are delivered.

SQL Code:

SELECT * FROM COURIER WHERE C_STATUS='DELIVERED';

Relational Algebra:

$\Pi_{<c.c_id>}(\rho(c, \sigma_{<c.c_status="delivered">} courier))$

Output:

The screenshot shows a PostgreSQL Query Editor window. The query is as follows:

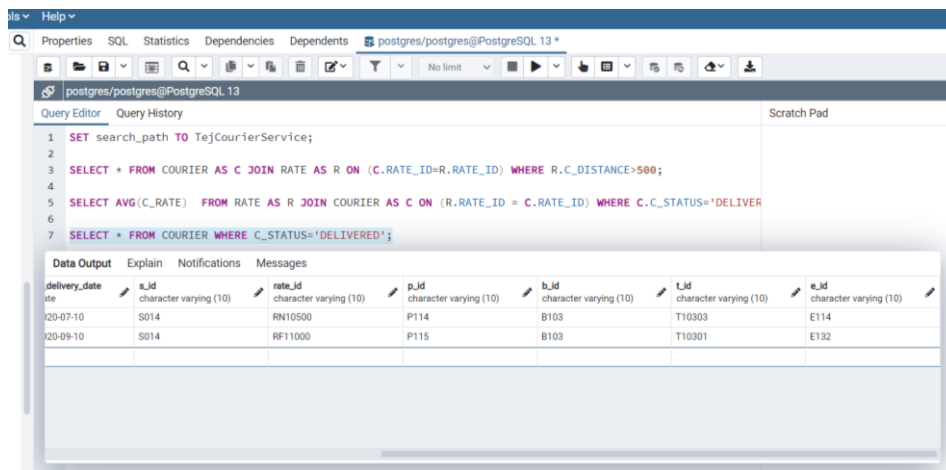
```

1 SET search_path TO TejCourierService;
2
3 SELECT * FROM COURIER AS C JOIN RATE AS R ON (C.RATE_ID=R.RATE_ID) WHERE R.C_DISTANCE>500;
4
5 SELECT AVG(C.RATE) FROM RATE AS R JOIN COURIER AS C ON (R.RATE_ID = C.RATE_ID) WHERE C.C_STATUS='DELIVERED';
6
7 SELECT * FROM COURIER WHERE C_STATUS='DELIVERED';

```

The Data Output tab shows the following result:

c_id	c_name	c_dispatch_date	c_status	c_delivery_date	s_id	rate_id
1 C005	MNO	2020-07-02	DELIVERED	2020-07-10	S014	RN10500
2 C006	MNO	2020-09-02	DELIVERED	2020-09-10	S014	RF11000



delivery_date	s_id	rate_id	p_id	b_id	t_id	e_id
20-07-10	S014	RN10500	P114	B103	T10303	E114
20-09-10	S014	RF11000	P115	B103	T10301	E132

5. To retrieve the couriers which have been dispatched.

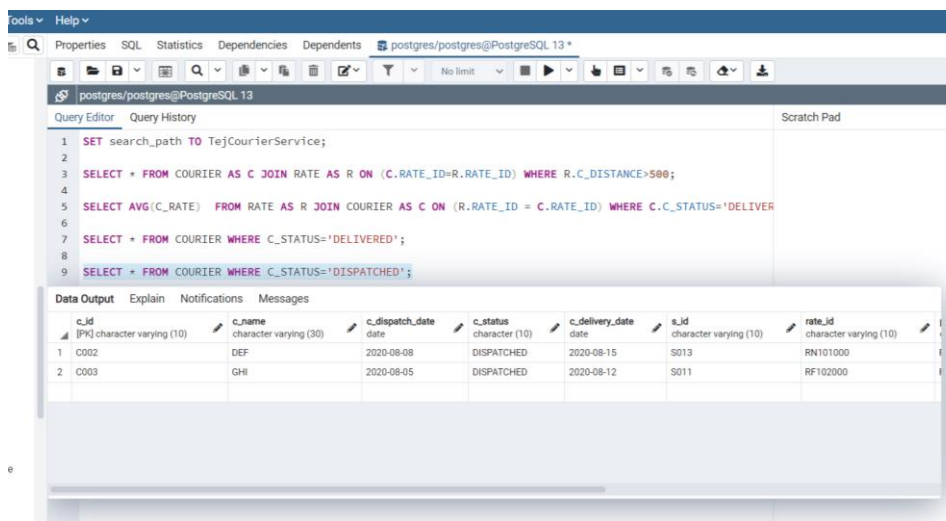
SQL Code:

```
SELECT * FROM COURIER WHERE C_STATUS='DISPATCHED';
```

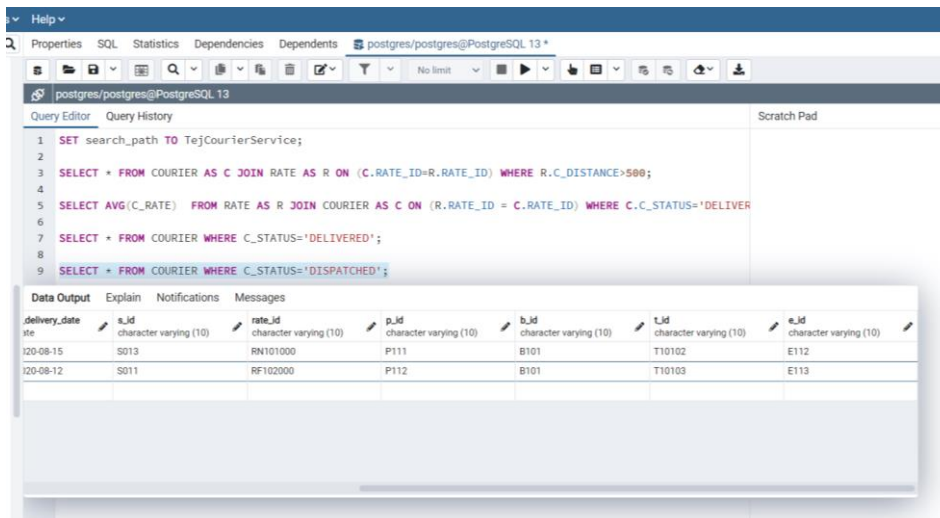
Relational Algebra:

$$\Pi_{\langle c.c_id \rangle}(\rho(c, \sigma_{\langle c.c_status = \text{"dispatched"} \rangle} \text{courier}))$$

Output:



c_id	c_name	c_dispatch_date	c_status	c_delivery_date	s_id	rate_id
C002	DEF	2020-08-08	DISPATCHED	2020-08-15	S013	RN101000
C003	GHI	2020-08-05	DISPATCHED	2020-08-12	S011	RF102000



6. To retrieve the Employee's name and his/her Designation.

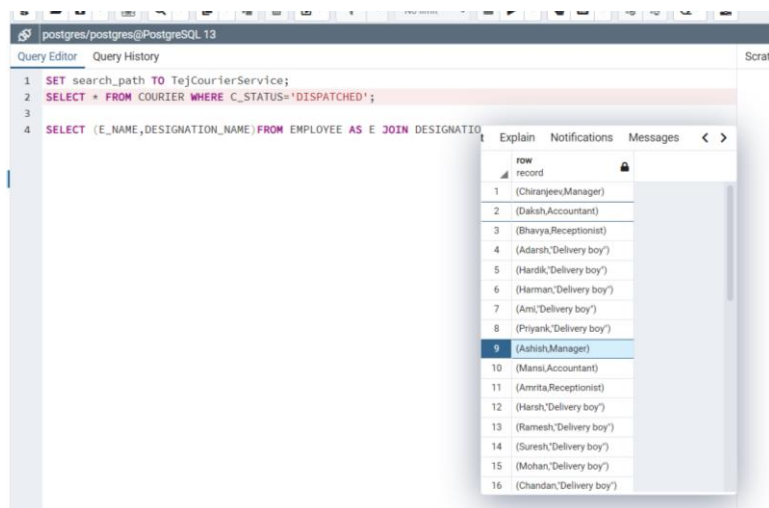
SQL Code:

```
SELECT (E_NAME, DESIGNATION_NAME) FROM EMPLOYEE AS E JOIN
DESIGNATION D ON (E.E_ID=D.E_ID);
```

Relational Algebra:

$$\Pi_{\langle e_name, designation_name \rangle} ((employee) \bowtie_{\langle employee.eid=designation.did \rangle} (designation))$$

Output:



7. To Retrieve payment mode and Transaction Details.

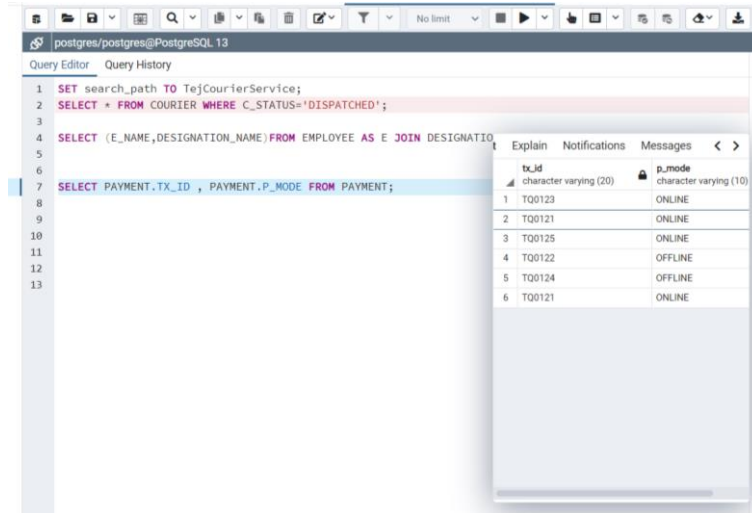
SQL Code:

```
SELECT PAYMENT.TX_ID , PAYMENT.P_MODE FROM PAYMENT;
```

Relational Algebra:

$$\Pi_{\langle p_mode, tx_id \rangle}(\text{payment})$$

Output:



The screenshot shows a PostgreSQL query editor with the following SQL code:

```
1 SET search_path TO TejCourierService;
2 SELECT * FROM COURIER WHERE C_STATUS='DISPATCHED';
3
4 SELECT (E_NAME,DESIGNATION_NAME)FROM EMPLOYEE AS E JOIN DESIGNATION
5
6
7 SELECT PAYMENT.TX_ID , PAYMENT.P_MODE FROM PAYMENT;
```

The output window displays the following data:

tx_id	p_mode
TQ0123	ONLINE
TQ0121	ONLINE
TQ0125	ONLINE
TQ0122	OFFLINE
TQ0134	OFFLINE
TQ0121	ONLINE

8. To retrieve info about courier id no C005 (Weight, Distance and type)

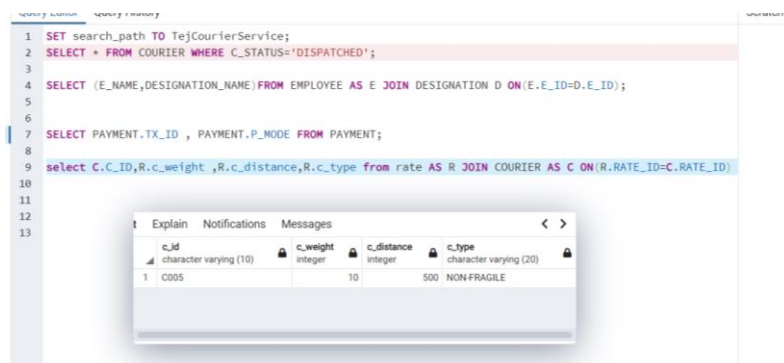
SQL Code:

```
select C.C_ID,R.c_weight ,R.c_distance,R.c_type from rate AS R JOIN COURIER
AS C ON(R.RATE_ID=C.RATE_ID) WHERE C.C_ID='C005';
```

Relational Algebra:

$$\Pi_{\langle c_id, r_c_weight, r_c_distance, r_c_type \rangle} \rho(c, (\sigma_{\langle c_id=C005 \rangle} \text{courier})) \bowtie_{\langle r_rate_id=c_rate_id \rangle} \rho(r, \text{rate})$$

Output:



The screenshot shows a PostgreSQL query editor with the following SQL code:

```
1 SET search_path TO TejCourierService;
2 SELECT * FROM COURIER WHERE C_STATUS='DISPATCHED';
3
4 SELECT (E_NAME,DESIGNATION_NAME)FROM EMPLOYEE AS E JOIN DESIGNATION D ON(E.E_ID=D.E_ID);
5
6
7 SELECT PAYMENT.TX_ID , PAYMENT.P_MODE FROM PAYMENT;
8
9 select C.C_ID,R.c_weight ,R.c_distance,R.c_type from rate AS R JOIN COURIER AS C ON(R.RATE_ID=C.RATE_ID)
```

The output window displays the following data:

c_id	c_weight	c_distance	c_type
C005	10	500	NON-FRAGILE

9. Total no. of fragile products sent for each of the year from 2016 to 2020

SQL Code:

```
SELECT COUNT (*) FROM COURIER AS C JOIN RATE AS R ON
(C.RATE_ID=R.RATE_ID) WHERE R.C_TYPE='FRAGILE' AND
C.C_DISPATCH_DATE BETWEEN '#2016-01-01#' AND '#2020-12-01#';
```

Relational Algebra:

$$r1 \leftarrow \rho(c, \sigma_{\langle c_dispatch_date \rangle 2016-01-01 \text{ AND } \langle c_dispatch_date \rangle 2016-12-31} \text{ courier})$$

$$r2 \leftarrow \rho(r, \sigma_{\langle c_type \rangle = 'Fragile'} \text{ rate})$$

$$\text{result} \leftarrow \mathcal{F}_{\text{COUNT} (*)} r1 \bowtie_{\langle c.rate_id=r.rate_id \rangle} r2$$

Output:

count	bigint
4	

10. To find payable amount of courier which has 10 kg or less weight.

SQL Code:

```
SELECT C.C_ID,R.C_RATE,R.C_WEIGHT from courier as c join rate as r on
(c.rate_id=r.rate_id) where r.c_weight<='10';
```

Relational Algebra:

$$\Pi_{c.c_id, R.c_rate, R.c_weight} ((\sigma_{\langle R.c_weight \rangle \leq 10} (\rho(c, \text{couriers}))) \bowtie_{\langle c.rate_id=R.rate_id \rangle} \rho(r, \text{rate}))$$

Output:

c_id	c_rate	c_weight	integer
C001	120	10	
C002	140	10	
C003	600	10	
C005	120	10	
C006	200	1	
C007	200	1	

11. Total no. of Employee working in Ahmedabad branch.

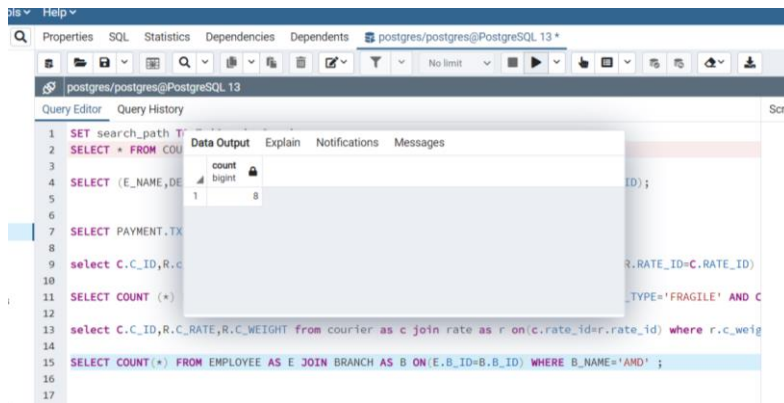
SQL Code:

```
SELECT COUNT(*) FROM EMPLOYEE AS E JOIN BRANCH AS B  
ON(E.B_ID=B.B_ID) WHERE B_NAME='AMD' ;
```

Relational Algebra:

$$\mathcal{F}_{\text{COUNT} (*)} ((\sigma_{\langle B_name=AMD \rangle} (\rho(E, \text{Employee}))) \bowtie_{\langle E.b_id=B.b_id \rangle} \rho(B, \text{Branch}))$$

Output:



The screenshot shows the PostgreSQL Query Editor with the following SQL query:

```
1 SET search_path TO 'public';  
2 SELECT * FROM COURIER;  
3  
4 SELECT (E_NAME,DE  
5  
6  
7 SELECT PAYMENT.TX  
8  
9 select C.C_ID,R.C  
10  
11 SELECT COUNT (*)  
12  
13 select C.C_ID,R.C_RATE,R.C_WEIGHT from courier as c join rate as r on(c.rate_id=r.rate_id) where r.c_weig  
14  
15 SELECT COUNT(*) FROM EMPLOYEE AS E JOIN BRANCH AS B ON(E.B_ID=B.B_ID) WHERE B_NAME='AMD' ;  
16  
17
```

The output window displays the result of the query:

count
8

12. To retrieve name of employees who are male and working in branch Mumbai.

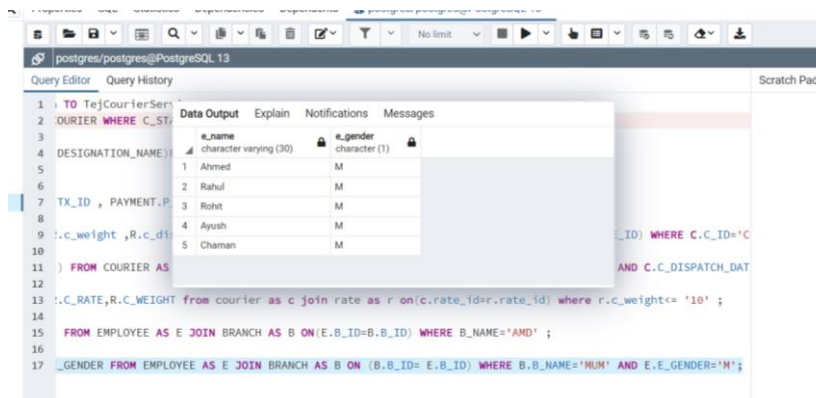
SQL Code:

```
SELECT E_NAME,E_GENDER FROM EMPLOYEE AS E JOIN BRANCH AS B  
ON (B.B_ID= E.B_ID) WHERE B.B_NAME='MUM' AND E.E_GENDER='M';
```

Relational Algebra:

$$\Pi_{e_name, e_gender} ((\sigma_{\langle B.b_name=mum \wedge E.e_gender=m \rangle} (\rho(e, \text{employee}))) \bowtie_{\langle B.bid=E.bid \rangle} \rho(B, \text{Branch}))$$

Output:



The screenshot shows the PostgreSQL Query Editor with the following SQL query:

```
1 TO TejCourierSer  
2 COURIER WHERE C_ST  
3  
4 DESIGNATION_NAME))  
5  
6  
7 TX_ID , PAYMENT.P  
8  
9 .C_weight ,R.c_di  
10  
11 FROM COURIER AS  
12  
13 .C_RATE,R.C_WEIGHT from courier as c join rate as r on(c.rate_id=r.rate_id) where r.c_weight <= '10' ;  
14  
15 FROM EMPLOYEE AS E JOIN BRANCH AS B ON(E.B_ID=B.B_ID) WHERE B_NAME='AMD' ;  
16  
17 GENDER FROM EMPLOYEE AS E JOIN BRANCH AS B ON (B.B_ID= E.B_ID) WHERE B.B_NAME='MUM' AND E.E_GENDER='M';
```

The output window displays the result of the query:

e_name	e_gender
Ahmed	M
Rahul	M
Rohit	M
Ayush	M
Chaman	M

13. To select name of customer who has sent courier between 1st August 2020 to 30 September 2020.

SQL Code:

```
SELECT FIRST_NAME, LAST_NAME FROM SENDER AS S JOIN COURIER
AS C ON(S.S_ID=C.S_ID) WHERE C.C_DISPATCH_DATE BETWEEN '#2020-
08-01#' AND '#2020-08-30#';
```

Relational Algebra:

$$\Pi_{\langle \text{first_name}, \text{last_name} \rangle} (\rho(S, (\sigma_{\langle \text{c.dispatch_date} \rangle 2020-08-01 \text{ and } \text{c.dispatch_date} < 2020-09-30} \text{sender})) \bowtie_{\langle \text{s.s_id} = \text{c.s_id} \rangle} \rho(\text{c}, \text{courier}))$$

Output:

first_name	last_name
1 RAHUL	OBEROI
2 BABITA	IYER
3 RAHUL	OBEROI
4 RAJ	MEHTA

14. To retrieve name of branch manager of Ahmedabad branch

SQL Code:

```
SELECT E_NAME FROM EMPLOYEE AS E JOIN DESIGNATION AS D
ON(E.E_ID=D.E_ID) WHERE D.DESIGNATION_NAME='Manager' AND
E.B_ID='B101';
```

Relational Algebra:

$$\Pi_{\text{e_name}} ((\sigma_{\langle \text{D.designation_name} = \text{Manager AND E.bid} = \text{B101} \rangle} (\rho(\text{e}, \text{employee})) \bowtie_{\langle \text{EE.e_id} = \text{D.e_id} \rangle} \rho(\text{D}, \text{Designation})))$$

Output:

The screenshot shows the PostgreSQL Query Editor with a query in the Query Editor tab. The query is:

```
1 SET search_path TO 'public';
2 SELECT * FROM COURIER;
3
4 SELECT (E_NAME, DE
5
6
7 SELECT PAYMENT.TX
8
9 select C.C_ID,R.C
10
11 SELECT COUNT (*)
12
13 select C.C_ID,R.C_RATE,R.C_WEIGHT from courier as c join rate as r on(c.rate_id=r.rate_id) where r.c_weig
14
15 SELECT COUNT(*) FROM EMPLOYEE AS E JOIN BRANCH AS B ON(E.B_ID=B.B_ID) WHERE B_NAME='AMD' ;
16
17 SELECT E_NAME,E_GENDER FROM EMPLOYEE AS E JOIN BRANCH AS B ON (B.B_ID= E.B_ID) WHERE B_NAME='MUM' AND E
18
19 SELECT FIRST_NAME, LAST_NAME FROM SENDER AS S JOIN COURIER AS C ON(S.S_ID=C.S_ID) WHERE C.C_DISPATCH_DATE
20
21 SELECT E_NAME FROM EMPLOYEE AS E JOIN DESIGNATION AS D ON(E.E_ID=D.E_ID) WHERE D.DESIGNATION_NAME='Manage
22
```

The Data Output tab shows the following data:

e_name
Chiranjeev

15. Retrieve the name of the employee who have salary more than 20,000 but less than 70,000

SQL Code:

```
SELECT E_NAME,SALARY FROM EMPLOYEE AS E JOIN DESIGNATION AS D ON(E.E_ID=D.E_ID) WHERE D.SALARY > 20000 AND D.SALARY < 70000;
```

Relational Algebra:

$$\Pi_{e_name, salary} ((\sigma_{<D.salary > 20000 \text{ AND } D.salary < 70000>} (\rho(e, employee)) \bowtie_{<E.E_id=D.e_id>} \rho(D, Designation))$$

Output:

The screenshot shows the PostgreSQL Query Editor with a query in the Query Editor tab. The query is:

```
1 SET search_path TO 'public';
2 SELECT * FROM COURIER;
3
4 SELECT (E_NAME, DE
5
6
7 SELECT PAYMENT.TX
8
9 select C.C_ID,R.C
10
11 SELECT COUNT (*)
12
13 select C.C_ID,R.C
14
15 SELECT COUNT(*) F
16
17 SELECT E_NAME,E_G
18
19 SELECT FIRST_NAME
20
21 SELECT E_NAME FR
22
23 SELECT E_NAME,SALARY FROM EMPLOYEE AS E JOIN DESIGNATION AS D ON(E.E_ID=D.E_ID) WHERE D.SALARY > 20000 AN
24
```

The Data Output tab shows the following data:

e_name	salary
Chiranjeev	50000
Daksh	30000
Bhavya	25000
Ashish	50000
Mansi	30000
Amrita	25000
Priya	50000
Dilp	30000
Zeel	25000
Gauri	50000
Simaran	30000
Nisha	25000
Champak	50000
Anish	30000
Nupur	25000

16. Find the courier ID whose Delivery status is FAILED.

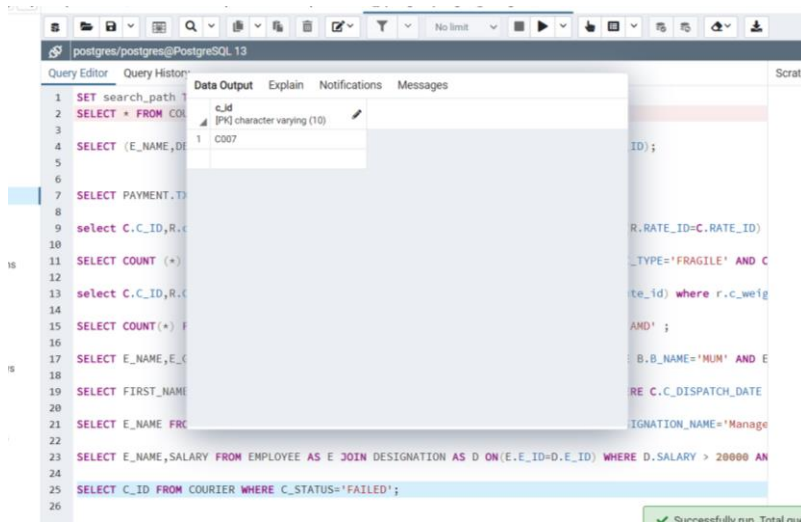
SQL Code:

```
SELECT C_ID FROM COURIER WHERE C_STATUS='FAILED';
```

Relational Algebra:

$$\Pi_{\langle c_id \rangle}(\sigma_{\langle c_status = \text{"failed"} \rangle}(\text{courier}))$$

Output:



17. Total number of couriers delivered in Maharashtra In Year 2020.

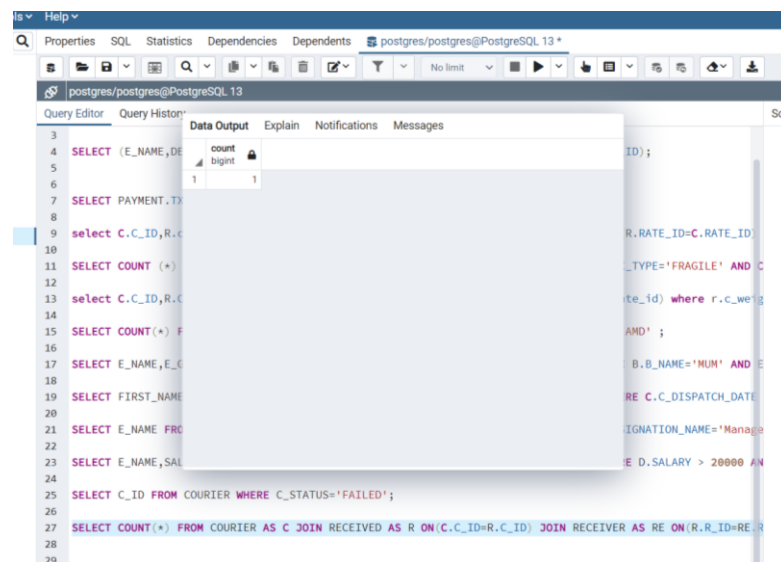
SQL Code:

```
SELECT COUNT(*) FROM COURIER AS C JOIN RECEIVER AS R
ON(R.R_ID=C.R_ID) WHERE R.R_STATE='MAHARASHTRA' AND
C.C_STATUS='DELIVERED' AND C_DELIVERY_DATE BETWEEN '#2020-01-
01#' AND '#2020-12-31#';
```

Relational Algebra:

$$r1 \leftarrow \rho(c, \sigma_{\langle c_delivery_date > 2020-01-01 \text{ AND } \langle c_delivery_date < 2020-12-31 \rangle} \text{ courier}))$$
$$r2 \leftarrow \rho(r, \sigma_{\langle r_state = \text{'Maharashtra'} \rangle} \text{ receiver}))$$
$$\text{result} \leftarrow \mathcal{F}_{\text{COUNT}(*)} r1 \bowtie_{\langle c_id = r_id \rangle} r2$$

Output:



The screenshot shows a PostgreSQL query editor with a SQL query in the Query Editor tab. The query is:

```
SELECT (E_NAME,DE  
SELECT PAYMENT.TX  
select C.C_ID,R.C  
SELECT COUNT (*)  
select C.C_ID,R.C  
SELECT COUNT(*) F  
SELECT E_NAME,E_C  
SELECT FIRST_NAME  
SELECT E_NAME FRO  
SELECT E_NAME,SAL  
SELECT C_ID FROM COURIER WHERE C_STATUS='FAILED';  
SELECT COUNT(*) FROM COURIER AS C JOIN RECEIVED AS R ON(C.C_ID=R.C_ID) JOIN RECEIVER AS RE ON(R.R_ID=RE.R_ID)
```

The Data Output tab shows the following result:

count	bigint
1	1

18. Retrieve courier's sender and receiver name whose STATUS is SUCCESSFUL.

SQL Code:

```
SELECT DISTINCT  
S.FIRST_NAME,S.LAST_NAME,R.FIRST_NAME,R.LAST_NAME FROM  
(SENDER AS S JOIN SENDTO AS SE ON(S.S_ID=SE.S_ID) JOIN RECEIVER AS  
R ON(SE.R_ID=R.R_ID) join courier as c on(c.s_id=s.s_id)) WHERE  
c.c_status='DELIVERED'
```

Relational Algebra:

$r1 \leftarrow \rho(c, \sigma_{<c.c_status='delivered'> courier})$

$r2 \leftarrow \rho(r, receiver)$

$r3 \leftarrow \rho(se, sendto)$

$r4 \leftarrow \rho(s, sender)$

$result \leftarrow \Pi_{<s.first_name, s.last_name, r.first_name, r.last_name>} r1 \bowtie_{<c.c_id=s.c_id>} r4 \bowtie_{<s.s_id=se.s_id>}$

$r3 \bowtie_{<se.r_id=r.r_id>} r2$

Output:

	first_name character varying (30)	last_name character varying (30)	first_name character varying (30)	last_name character varying (30)
1	SONALIKA	BHIDE	HIMANSHU	SHARMA
2	SONALIKA	BHIDE	SIDHARTH	GUPTA

19. Select name of customers whose mode of payment is offline and courier status is in transit.

SQL Code:

```
SELECT FIRST_NAME, LAST_NAME FROM SENDER AS S JOIN COURIER AS C ON (S.S_ID=C.S_ID) JOIN PAYMENT AS P ON (C.P_ID=P.P_ID) WHERE P.P_MODE='OFFLINE' AND C.C_STATUS='DISPATCHED';
```

Relational Algebra:

$$\Pi_{\langle s.\text{first_name}, s.\text{last_name} \rangle} \rho(s, (\sigma_{\langle p.\text{p_mode}="offline", c.\text{c_status}="dispatched" \rangle} \text{sender}) \bowtie_{\langle s.\text{sid}=c.\text{sid} \rangle} \rho(c, \text{courier}) \bowtie_{\langle c.\text{p_id}=p.\text{p_id} \rangle} \rho(p, \text{payment}))$$

Output:

	first_name character varying (30)	last_name character varying (30)
1	BABITA	IYER
2	RAHUL	OBEROI

20. Select all the courier whose cost is greater than Rs 200 and less than Rs 500 & mode of payment is online.

SQL Code:

```
SELECT C_ID, P.P_AMOUNT FROM COURIER AS C JOIN PAYMENT AS P ON (C.P_ID=P.P_ID) WHERE P.P_AMOUNT < '500' AND P.P_AMOUNT > '100' AND P.P_MODE='ONLINE';
```

Relational Algebra:

$$\Pi_{\langle c_id, p_id \rangle} (p, (\sigma_{\langle p_amount > 200 \text{ AND } p_p_amount < 500} (p_p_mode = "online") \bowtie_{\langle c_p_id = p_id \rangle} (c, courier)))$$

Output:

29 SELECT DISTINCT S.FIRST_NAME,S.LAST_NAME,R.FIRST_NAME,R.LAST_NAME FROM (SENDER AS S JOIN SENDTO AS SE ON (S.S_ID=SE.S_ID))

30

31 SELECT FIRST_NAME, LAST_NAME FROM SENDER AS S JOIN COURIER AS C ON (S.S_ID=C.S_ID) JOIN PAYMENT AS P ON (C.P_ID=P.P_ID)

32

33 SELECT C_ID,P.P_AMOUNT FROM COURIER AS C JOIN PAYMENT AS P ON (C.P_ID=P.P_ID) WHERE P.P_AMOUNT < '500' AND P.P.P_MODE = 'online'

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

Data Output

Explain

Notifications

Messages

	c_id	p_amount
	character varying (10)	integer
1	C001	120
2	C002	140
3	C006	200
4	C007	200