Courier Shipping Management System

- Basic queries with conditions and joins:
- 1) Get all couriers for a particular customer within a given date range.

Relational Query:

 σ (Customer_ID = '23110100002' AND Date >= '08-11-2023' AND Date <= '08-12-2023')(Courier)

SQL Query & Output:

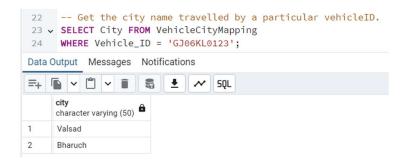


2) Get the city name travelled by a particular vehicleID. (Ex. GJ06KL0123):

Relational Query:

 $\pi_{\text{city}}(\sigma_{\text{vehicle_id}} = \text{"GJ06KL0123"})$ (vehiclecitymapping))

SQL Query & Output:



3) Get the lost courier within a given date range.

Relational Query:

σ (status = "Lost" AND "2024-01-01" <= date AND date <= "2024-01-10") (courier)

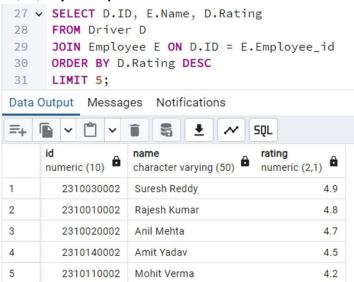


4) Get the top 5 most rated drivers.

Relational Query:

 $\pi_{d.id, e.name, d.rating}(\rho(d,driver) \bowtie (d.id=e.employee_id) \rho(e,employee))$

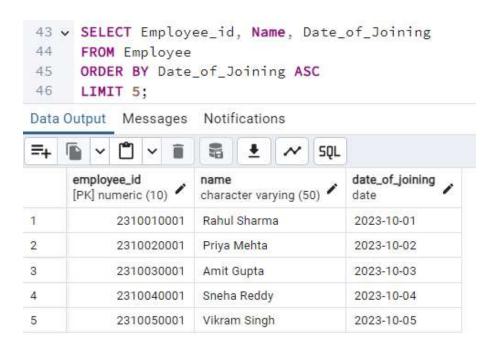
SQL Query & Output:



5) Find out the top 5 employees that have been with the company the longest.

Relational Query:

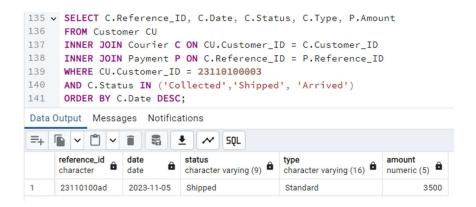
π employee_id, name, date_of_joining (employee)



6) Get Active Couriers of a particular customer

Relational Query:

```
\pi_c reference_id, c . date, c . status, c . type, p . amount ( \sigma_{cu} . customer_id = 23110100003 AND (c . status = "Collected" OR c . status = "Shipped" OR c . status = "Arrived") (\rho(Cu,customer) \bowtie_{cu} customer_id = c . customer_id \rho(C,courier) \bowtie_{c} reference_id = p . reference_id \rho(P,payment)))
```



Queries with aggregation in SELECT and aggregated conditions in HAVING clause:

7) Get the total weight of couriers sent from a particular city.

Relational Query:

 π SUM (c.weight) \rightarrow total_weight (\mathcal{F} SUM (c.weight) (σ b . city = "Valsad" (ρ (c,courier) \bowtie c . branch_id = b . branch_id ρ (b,branch))))

SQL Query & Output:

```
-- Get the total weight of couriers sent from a particular city.

SELECT SUM(C.Weight) AS Total_Weight

FROM Courier C

JOIN Branch B ON C.Branch_ID = B.Branch_ID

WHERE B.City = 'Valsad';

Data Output Messages Notifications

The Couriers sent from a particular city.

SELECT SUM(C.Weight) AS Total_Weight

FROM Courier C

1 JOIN Branch B ON C.Branch_ID = B.Branch_ID

WHERE B.City = 'Valsad';

Data Output Messages Notifications

The Couriers sent from a particular city.

SELECT SUM(C.Weight) AS Total_Weight

FROM Courier C

1 JOIN Branch B ON C.Branch_ID = B.Branch_ID

1 Valsad';

Data Output Messages Notifications
```

8) Get the top 5 most frequent destination cities for couriers.

Relational Query:

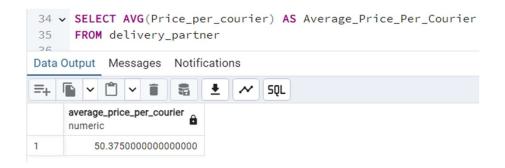
 π b. city, COUNT (*) \rightarrow courier_count(b.city \mathcal{F}_{COUNT} (*)(ρ (C,courier) \bowtie c. to_branch = b. branch_id ρ (b,branch)))



9) Get the average price per courier for all delivery partners.

Relational Query:

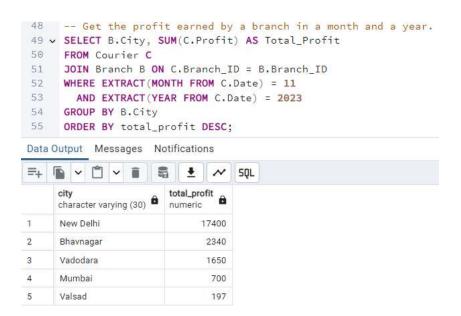
 π AVG (price_per_courier) \rightarrow average_price_per_courier(\mathcal{F} AVG (price_per_courier) (delivery_partner)) SQL Query & Output:



10) Get the profit earned by a branch in a particular month and a year.

Relational Query:

 π b . city, SUM (c.profit) \rightarrow total_profit (b.city \mathcal{F} SUM (c.profit)(σ month = 11 AND year = 2023(ρ (C,courier) \bowtie c . branch_id ρ (b,branch))))

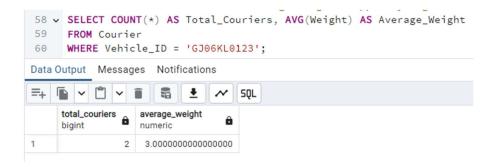


11) Get the total courier and average weight shipped by a given Vehicle. (Ex. GJ06KL0123)

Relational Query:

$$\begin{split} &\pi_{\text{COUNT (*)}} \rightarrow \text{total_couriers, AVG (weight)} \rightarrow \text{average_weight} \\ &\left(\mathcal{F}_{\text{COUNT (*),AVG (weight)}}(\sigma_{\text{vehicle_id}} = \text{"GJ06KL0123"(courier))))} \end{aligned}$$

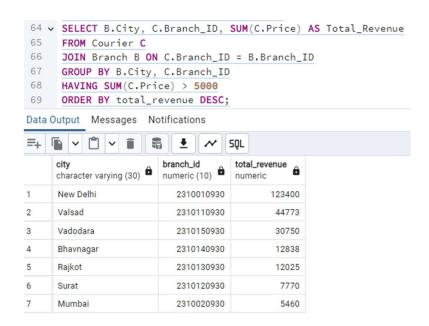
SQL Query & Output:



12) Get the total revenue generated by each branch, showing only branches where revenue exceeds 5000 Rupees.

Relational Query:

 π_b . city, c . branch_id, SUM (c.price) \rightarrow total_revenue σ_{SUM} (c.price) > 5000(b.city, c.branch_id \mathcal{F}_{SUM} (c.price)($\rho(C,courier) \bowtie_{c . branch_id} = b . branch_id = p(b,branch)))$

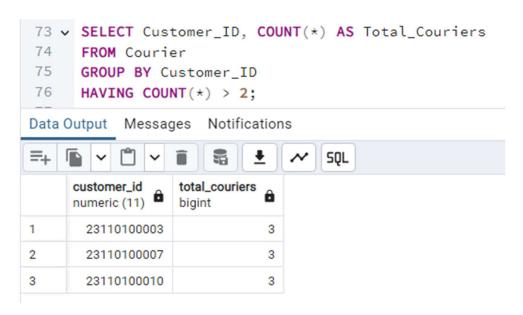


13) Get the number of couriers sent by each customer and show only those who have sent more than 2 couriers.

Relational Query:

 $\pi_{\text{customer_id}}$, COUNT (*) \rightarrow total_couriers (σ_{COUNT} (*) > 2 (customer_id $\mathcal{F}_{\text{COUNT}}$ (*) (Courier)))

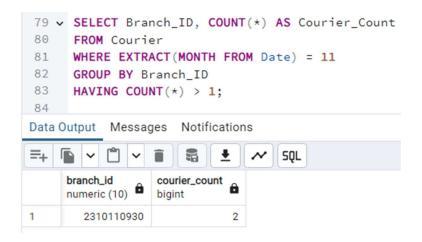
SQL Query & Output:



14) Find branches that have handled more than 1 courier in a specific month.

Relational Query:

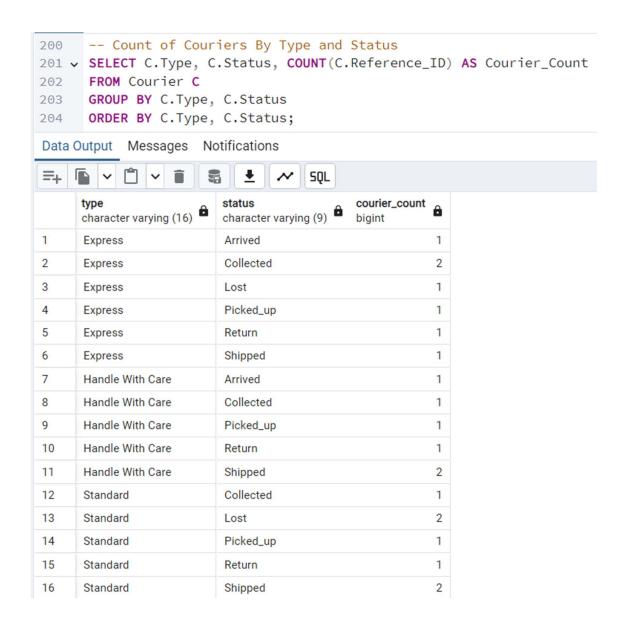
 π branch_id, COUNT (*) \rightarrow courier_count(σ COUNT (*) > 1(branch_id \mathcal{F} COUNT (*) (σ month = 11 (COURT ()))) SQL Query & Output:



15) Count of Couriers by Type and Status:

Relational Query:

 $\pi_{c.type, c.status}$, COUNT (reference_id) \rightarrow courier_count(type, status $\mathcal{F}_{count}(c.reference_id)$))

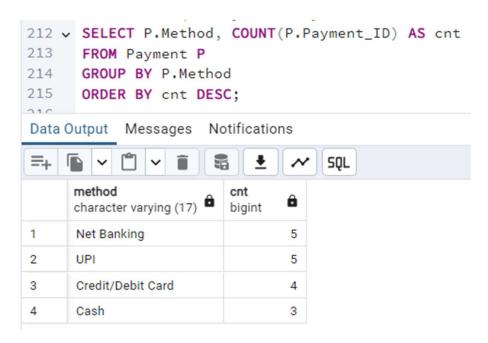


16) Most Frequently Used Payment Methods.

Relational Query:

 π_{p} . method, COUNT (p.payment_id) \rightarrow cnt(p.method \mathcal{F} COUNT (p.payment_id)(ρ (p,payment)))

SQL Query & Output:

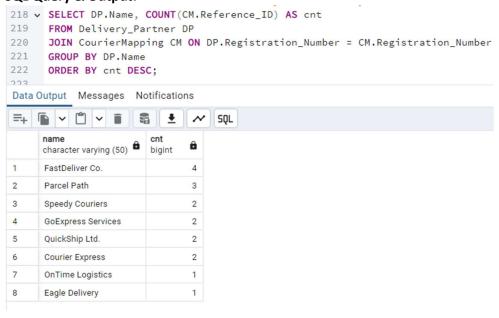


17) Total Number of Couriers Handled by Each Delivery Partner

Relational Query:

 $\pi_{\text{dp.name, COUNT (cm.reference_id)}} \rightarrow \text{cnt} \\ \left(\text{name} \mathcal{F}_{\text{COUNT (cm.reference_id)}} \right) \\ \left(\rho(\text{dp,delivery_partner}) \right) \\ \left(\rho($

 \bowtie dp . registration_number = cm . registration_number ρ (cm,couriermapping)))

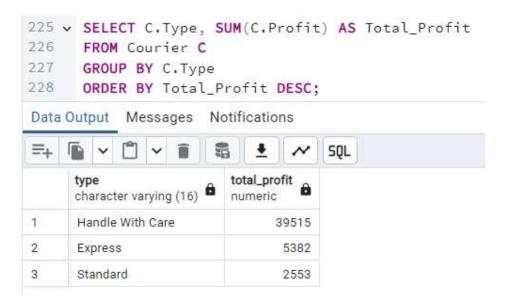


18) Most Profitable Courier Types

Relational Query:

 $\pi_{c.type}$, SUM (c.profit) \rightarrow total_profit(c.type \mathcal{F} SUM (c.profit) (ρ (C,courier)))

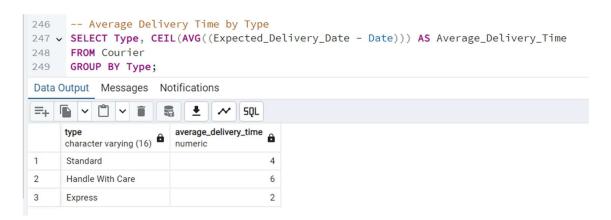
SQL Query & Output:



19) Average Delivery Time by Type.

Relational Query:

 $\pi_{\text{type}, \text{AVG}}$ (Expected_Delivery_Date - Date) \rightarrow Average_Delivery_Time(type \mathcal{F}_{AVG} (Expected_Delivery_Date - Date) (Courier))



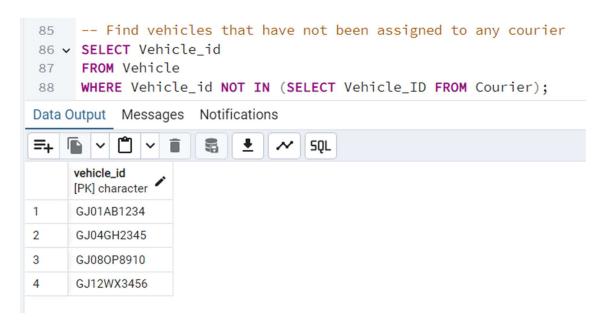
• Queries having nested queries:

20) Find vehicles that have not been assigned to any courier.

Relational Query:

 $\pi_{\text{vehicle_id}}$ (Vehicle) EXCEPT $\pi_{\text{vehicle_id}}$ (courier)

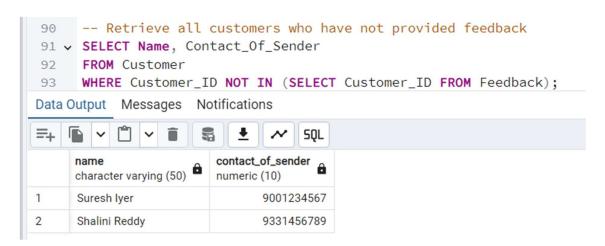
SQL Query & Output:



21) Retrieve all customers who have not provided feedback.

Relational Query:

 $\pi_{Name, Contact_Of_Sender}(\sigma_{customer_ID \ EXCEPT \ \pi_{Customer_ID}(Feedback)}(Customer))$



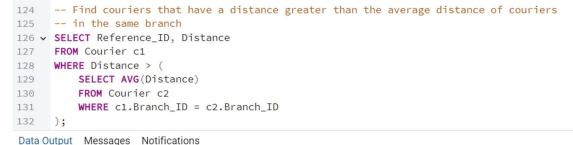
• Queries having aggregation in nested queries:

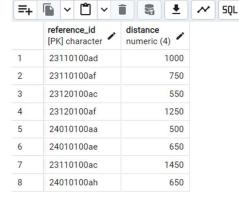
22) Find couriers that have a distance greater than the average distance of couriers in the same branch

Relational Query:

 $\pi \text{ reference_id, distance} \left(\sigma \text{ distance} \right) \left(\pi_{\text{AVG (distance)}} \left(\sigma_{\text{c1. branch_id}} \left(\rho(\text{c2. branch_id}) \right) \right) \right) \left(\rho(\text{c1,courier}) \right) \right) \left(\rho(\text{c1,courier}) \right) \right)$

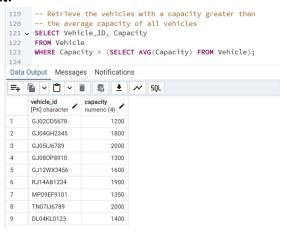
SQL Query & Output:





23) Retrieve the vehicles with a capacity greater than the average capacity of all vehicles **Relational Query:**

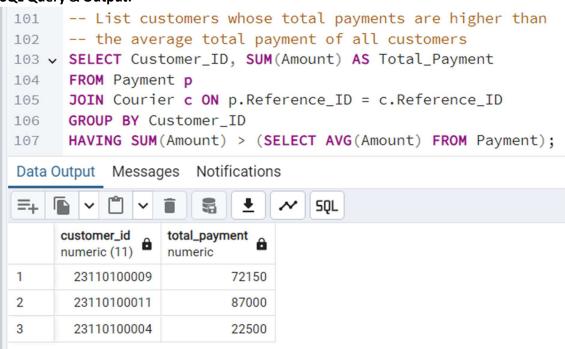
 π vehicle_id, capacity (σ capacity > (π AVG (capacity) (\mathcal{F} AVG (capacity) (vehicle))) (vehicle))



24) List customers whose total payments are higher than the average total payment of all customers

Relational Query:

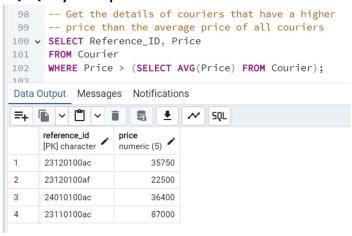
```
\pi customer_id, SUM (amount) \rightarrow total_payment(\sigma SUM (amount) \rightarrow total_payment(\sigma SUM (amount) \rightarrow total_payment) (\sigma SUM (amount) (\sigma SUM (\sigma SUM
```



25) Get the details of couriers that have a higher price than the average price of all couriers

Relational Query:

 $\pi_{\text{reference_id}}$, price $\{\sigma_{\text{price}} > (\pi_{\text{AVG (price)}}(\mathcal{F}_{\text{AVG (price)}}(\text{courier}))\}$ (Courier))



• Correlated queries:

26) Find names of delivery partners who have delivered at least 3 couriers

Relational Query:

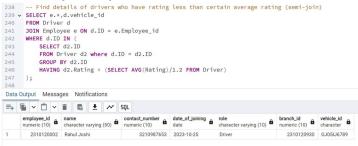
```
\pi dp . name(\sigma dp . registration_number SEMI-INTERSECTION<dp.registration_number=cm.registration_number>
(\pi_{\text{cm.registration_number}}(\sigma_{\text{count}(\text{cm.reference\_id})})))))))
(\rho(\text{dp,delivery\_partner})))
SQL Query & Output:
```

228 -- Find names of delivery partner who have delivered at least 3 courier 229 v SELECT dp.Name 230 FROM Delivery_Partner dp 231 WHERE dp.Registration_Number IN (SELECT cm.Registration_Number 232 FROM CourierMapping cm where dp.Registration_Number = cm.Registration_Number 233 234 GROUP BY cm.Registration_Number HAVING COUNT(cm.Reference_ID) >= 3 235 236); Data Output Messages Notifications SQL character varying (50) FastDeliver Co. 1 2 Parcel Path

27) Find details of drivers who have rating less than certain average rating.

Relational Query:

```
\begin{split} \pi_{e,*,d.vehicle\_id}(\sigma_{d:id} = \pi_{d2.id}(\sigma_{d2.rating < (\pi_{AVG (rating)/1.2} (\mathcal{F}_{AVG (rating)} (driver)))(d2.id\mathcal{F}(\sigma_{d:id} = d2.id(\rho(d2,driver)))))} \\ (\rho(d,driver) \bowtie_{d:id} = e.employee\_id \ \rho \ (e,employee))) \end{split}
```



• Queries with division operation:

28) Retrieve driver details who have driven couriers of all types.

Relational Query:

```
\begin{split} &\pi_{\text{Employee.*}}(\sigma_{\text{e.Employee\_ID= d.ID}}\left(\rho(e,\text{Employee})\bowtie \right. \\ &\left. (\pi_{\text{ID}}(d) \text{ EXCEPT } \pi_{\text{ID}}\left(r2\right) (\rho\left(r2,\left(\pi_{\text{d.ID->ID, cour.type -> type}}(\rho\left(d,\text{Driver}\right) \text{ X } (\pi_{\text{Type}} \rho(\text{cour,Courier})))\right) \right. \\ &\left. \text{EXCEPT } \pi_{\text{d.ID, c.Type}}(\rho(d,\text{Driver})\bowtie <_{\text{d.vehicle\_id=c.vehicle\_id>}} \rho\left(c,\text{Courier})))))))) \right. \end{split}
```

SQL Query & Output:

```
143
     -- Retrieve driver details who have driven couriers of all types
144 v SELECT *
145
     FROM Employee AS e
146
     WHERE e.employee_id = (
          SELECT DISTINCT d.id
147
148
          FROM Driver d
149
          WHERE d.ID NOT IN (
150
              SELECT ID
151
              FROM (
                  SELECT d.ID AS ID, cour.Type AS Type
152
153
                  FROM Driver d
154
                  CROSS JOIN (SELECT DISTINCT Type FROM Courier) AS cour
155
                  EXCEPT
156
                  SELECT d.ID, c.Type
                  FROM Driver d
157
                  JOIN Courier c ON d.Vehicle_ID = c.Vehicle_ID
158
159
              ) AS r2
160
161
    );
```

Data Output Messages Notifications ✓ SQL date_of_joining employee_id branch_id contact_number numeric (10) [PK] numeric (10) character varying (50) character varying (10) numeric (10) date 2310030002 Suresh Reddy 8765432105 2023-10-25 2310030930

29) Find branches that have managed couriers with all possible status.

Relational Query:

```
\sigma_{bcv.branch\_ID} = b.branch\_ID (\rho(b,Branch))
(\pi_{Branch\_ID}(BCV) EXCEPT \pi_{Branch\_ID}(r2)(\rho (r2, (\pi_{b.branch\_id->branch\_id, cour.status -> status}(\rho (r2)))
(b,Branch) X (\pi_{Status} \rho(cour,Courier))) EXCEPT \pi_{bcv,branch} ID, bcv,Status(BCV)))))
SQL Query & Output:
        -- Find branches that have managed couriers with all possible status
 164 V CREATE VIEW BCV AS
        SELECT b.Branch_ID, c.Status
 165
 166
        FROM Branch b
        JOIN Courier c ON b.Branch_ID = c.Branch_ID;
 167
 168
 169 v SELECT * FROM branch AS b
 170
        WHERE b.branch_id = (
             SELECT DISTINCT bcv.Branch_ID FROM BCV
 171
 172
             WHERE Branch_ID NOT IN (
                  SELECT Branch_ID
 173
                  FROM (
 174
                       SELECT b.Branch_ID AS Branch_ID, cour.Status AS Status
 175
                       FROM Branch b
 176
 177
                       CROSS JOIN (SELECT DISTINCT Status FROM Courier ) AS cour
                       EXCEPT
 178
 179
                       SELECT bcv.Branch_ID, bcv.Status
 180
                       FROM BCV
 181
                  ) AS r2
             )
 182
 183
        );
 Data Output Messages
                         Notifications
```

 30) Find Promotion Companies Details which are promoted by all the vehicles.

 $\pi_{\text{vpm.vehicle id,vpm.promotion id}}(\rho(\text{vpm,VehiclePromotionMapping}))))))$

Relational Query:

```
\begin{split} &\sigma_{promotion\_ID=promotion\_ID} \text{ (promotion} \bowtie \\ &(\pi_{promotion\_ID} \text{(VehiclePromotionMapping) EXCEPT } \pi_{promotion\_ID} \text{ (r2)} \text{ ($r2$, ($\pi_{v.vehicle\_id->vid, vpm.promotion\_id}$($\rho$ (v,Vehicle) X ($\rho$(vpm,VehiclePromotionMapping)))) EXCEPT \end{split}
```

```
-- Find Promotion Companies Details which are promoted by all the vehicles
177 ∨ SELECT * FROM promotion
      WHERE promotion_id = (
178
          SELECT DISTINCT promotion_id
179
          FROM VehiclePromotionMapping
180
          WHERE promotion_id NOT IN (
181
              SELECT promotion_id
182
               FROM (
183
                   SELECT v.Vehicle_id AS vid, vpm.promotion_id
184
185
                   FROM Vehicle v
                   CROSS JOIN (SELECT * FROM VehiclePromotionMapping) AS vpm
186
187
                   SELECT vpm.Vehicle_id, vpm.promotion_id
188
                   FROM VehiclePromotionMapping AS vpm
189
190
              ) AS r2
191
192
      );
102
Data Output Messages
                     Notifications
    SQL
=+
     promotion_id
                   price
                               company_name
                               character varying (50)
     [PK] character
                   numeric (7)
     231101abcd
1
                         15000
                               Amul
```

Views:

• We have created One View to use it in query.

