

Base Queries(BQ) and Analytical Queries(AQ)

BQ1:

Location/Sales class summary for job quantity and amount (revenue/costs)

SELECT

w_job_f.location_id,
location_name,
w_job_f.sales_class_id,
sales_class_desc,
w_time_d.time_year,
w_time_d.time_month,
base_price,
SUM(Quantity_ordered) AS sum_quantity_ordered,
SUM(Quantity_ordered*Unit_price)

FROM

w_job_f,
w_location_d,
w_sales_class_d,
w_time_d

WHERE

w_location_d.location_id = w_job_f.location_id
AND w_sales_class_d.sales_class_id = w_job_f.sales_class_id
AND w_time_d.time_id = w_job_f.contract_date

GROUP BY

w_job_f.location_id,
location_name,
w_job_f.sales_class_id,
sales_class_desc,
base_price,

w_time_d.time_year,

w_time_d.time_month

ORDER BY

location_id

The screenshot displays the pgAdmin 4 interface. On the left, the 'Databases (8)' tree shows the 'public' schema with various objects. The main pane shows a SQL query in the 'Query' editor, which is a complex aggregation query. The query selects columns from 'w_job_f.location_id', 'w_job_f.location_name', 'w_job_f.sales_class_id', 'sales_class_desc', 'w_time_d.time_year', 'w_time_d.time_month', 'base_price', and 'SUM(Quantity_ordered) AS sum_quantity_ordered, SUM(Quantity_ordered*base_price)'. It includes a 'FROM' clause with 'w_job_f', 'w_location_d', 'w_sales_class_d', and 'w_time_d'. The 'WHERE' clause filters for 'w_location_d.location_id = w_job_f.location_id' and 'w_sales_class_d.sales_class_id = w_job_f.sales_class_id'. The 'GROUP BY' clause lists 'w_job_f.location_id', 'w_location_name', 'w_job_f.sales_class_id', 'sales_class_desc', 'base_price', 'w_time_d.time_year', and 'w_time_d.time_month'. The 'ORDER BY' clause is 'location_id'.

The 'Data Output' pane shows the results of the query, which are 10 rows of data. The columns are: location_id, location_name, sales_class_id, sales_class_desc, time_year, time_month, base_price, sum_quantity_ordered, and sum. The data is as follows:

location_id	location_name	sales_class_id	sales_class_desc	time_year	time_month	base_price	sum_quantity_ordered	sum
1	New York	1	Debit Smart	2013	3	1.6000	613000	931740.00
2	New York	1	Debit Smart	2013	5	1.6000	537900	860702.00
3	New York	1	Debit Smart	2013	6	1.6000	346500	537075.00
4	New York	1	Debit Smart	2013	8	1.6000	894200	1341300.00
5	New York	1	Debit Smart	2013	9	1.6000	973000	1449770.00
6	New York	1	Debit Smart	2013	10	1.6000	1466000	2463440.00
7	New York	1	Debit Smart	2013	12	1.6000	325600	488960.00
8	New York	1	Debit Smart	2014	1	1.6000	1727400	2563872.00
9	New York	1	Debit Smart	2014	2	1.6000	76400	122240.00
10	New York	1	Debit Smart	2014	4	1.6000	2422100	3907476.00

Total rows: 1000 of 1368 Query complete 00:00:00.168

Successfully run. Total query runtime: 168 msec. 1368 rows affected.

BQ2:**Location invoice revenue summary (revenue/costs)**

```
WITH Inv_Rev_SummaryCTE AS (  
  SELECT  
    w_job_f.job_id,  
    w_job_f.location_id,  
    location_name,  
    unit_price AS Job_Unit_Price,  
    quantity_ordered AS Job_Order_Quantity,  
    w_time_d.time_year AS Year_Contract_Date,  
    w_time_d.time_month AS Month_Contract_Date,  
    SUM(w_invoiceline_f.invoice_amount) AS Sum_Invoice_Amount,  
    SUM(w_invoiceline_f.invoice_quantity) AS Sum_Invoice_Quantity  
  FROM  
    w_job_f,  
    w_location_d,  
    w_time_d,  
    w_invoiceline_f  
  WHERE  
    w_job_f.location_id = w_location_d.location_id  
    AND w_job_f.contract_date = w_time_d.time_id  
    AND w_job_f.location_id = w_invoiceline_f.location_id  
  GROUP BY  
    w_job_f.job_id,  
    w_job_f.location_id,  
    location_name,  
    unit_price,  
    quantity_ordered,
```

```

w_time_d.time_year,
w_time_d.time_month
)

```

```

SELECT
job_id,
location_id,
location_name,
Job_Unit_Price,
Job_Order_Quantity,
Year_Contract_Date,
Month_Contract_Date,
Sum_Invoice_Amount,
Sum_Invoice_Quantity
FROM
Inv_Rev_SummaryCTE
ORDER BY
location_id;

```

The screenshot shows the pgAdmin 4 interface. The left pane displays the database structure, including the 'public' schema and the 'Inv_Rev_SummaryCTE' table. The right pane shows the SQL query and its results.

Query:

```

1 WITH Inv_Rev_SummaryCTE AS (
2   SELECT
3     w_job_f.job_id,
4     w_job_f.location_id,
5     location_name,
6     unit_price AS Job_Unit_Price,
7     quantity_ordered AS Job_Order_Quantity,
8     w_time_d.time_year AS Year_Contract_Date,
9     w_time_d.time_month AS Month_Contract_Date,
10    SUM(w_invoice_line_f.invoice_amount) AS Sum_Invoice_Amount,
11    SUM(w_invoice_line_f.invoice_quantity) AS Sum_Invoice_Quantity
12  FROM
13    w_job_f,
14    w_location_d,
15    w_time_d,
16    w_invoice_line_f
17  WHERE
18    w_job_f.location_id = w_location_d.location_id
19    AND w_job_f.contract_date = w_time_d.time_id
20    AND w_job_f.location_id = w_invoice_line_f.location_id
21  GROUP BY
22    w_job_f.job_id,
23    w_job_f.location_id,
24    location_name,
25    unit_price,
26    quantity_ordered,
27    w_time_d.time_year,
28    w_time_d.time_month

```

Data Output:

job_id	location_id	location_name	job_unit_price	job_order_quantity	year_contract_date	month_contract_date	sum_invoice_amount	sum_invoice_quantity
1	341466	New York	1.98	144700	2013	12	106250909.00	102381191
2	341464	New York	0.77	677000	2013	4	106250909.00	102381191
3	342229	New York	0.75	786200	2013	4	106250909.00	102381191
4	340122	New York	0.79	183800	2014	1	106250909.00	102381191
5	341456	New York	0.77	601700	2013	9	106250909.00	102381191
6	341454	New York	0.78	559000	2013	9	106250909.00	102381191
7	341451	New York	1.47	1192500	2014	11	106250909.00	102381191
8	340479	New York	1.47	1174000	2013	12	106250909.00	102381191
9	341447	New York	0.77	666200	2014	1	106250909.00	102381191
10	342349	New York	0.79	173700	2014	4	106250909.00	102381191

Results: 1000 of 2569. Query complete 00:00:07.132. Successfully run. Total query runtime: 7 secs 132 msec. 2569 rows affected.

BQ3:**Location subjob cost summary (revenue/costs)**

WITH Loc_Subjob_SummaryCTE AS (

SELECT

w_job_f.job_id,

w_job_f.location_id,

location_name,

w_time_d.time_year AS Year_Contract_Date,

w_time_d.time_month AS Month_Contract_Date,

SUM(cost_labor) AS Sum_Labor_cost,

SUM(cost_material) AS Sum_material_cost,

SUM(machine_hours*rate_per_hour) AS Sum_machine_cost,

SUM(cost_overhead) AS Sum_overhead_cost,

SUM(cost_labor+ cost_material+ (machine_hours*rate_per_hour)+ cost_overhead) AS
Sum_Total_Cost,

SUM(quantity_produced) AS Sum_Quantity_Produced,

SUM(cost_labor+ cost_material+ (machine_hours*rate_per_hour)+ cost_overhead)/
SUM(quantity_produced) AS Unit_Cost

FROM

w_job_f

JOIN w_location_d ON w_location_d.location_id = w_job_f.location_id

JOIN w_time_d ON w_time_d.time_id = w_job_f.contract_date

JOIN w_sub_job_f ON w_sub_job_f.job_id = w_job_f.job_id

JOIN w_machine_type_d ON w_machine_type_d.machine_type_id =
w_sub_job_f.machine_type_id

GROUP BY

w_job_f.job_id,

w_job_f.location_id,

```
        location_name,  
        time_year,  
        time_month  
ORDER BY  
        job_id ASC  
)
```

```
SELECT  
    job_id,  
    location_id,  
    location_name,  
    Year_Contract_Date,  
    Month_Contract_Date,  
    Sum_Labor_cost,  
    Sum_material_cost,  
    Sum_machine_cost,  
    Sum_overhead_cost,  
    Sum_Total_Cost,  
    Sum_Quantity_Produced,  
    Unit_Cost  
FROM  
    Loc_Subjob_SummaryCTE
```

pgAdmin 4

File Object Tools Help

Browser

- Databases (8)
 - jobDB
 - Tables (3)
 - diagnosedtrauma
 - patient
 - patlog
 - Columns (6)
 - diagnode
 - diagnose
 - diagnosetype
 - diagnosetypeid
 - diagnosetypeid
 - diagnosetypeid
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
 - Sequences
 - Columns (4)
 - diagnode
 - diagnose
 - diagnosetype
 - diagnosetypeid

Query

Query History

```
1 WITH Loc_Subjob_SummaryCTE AS (  
2 SELECT  
3 w_job_f.job_id,  
4 w_job_f.location_id,  
5 location_name,  
6 w_time_d.time_year AS Year_Contract_Date,  
7 w_time_d.time_month AS Month_Contract_Date,  
8 SUM(cost_labor) AS Sum_Labor_cost,  
9 SUM(cost_material) AS Sum_material_cost,  
10 SUM(machine_hours*rate_per_hour) AS Sum_machine_cost,  
11 SUM(cost_overhead) AS Sum_overhead_cost,  
12 SUM(cost_labor+cost_material+ (machine_hours*rate_per_hour)+ cost_overhead ) AS Sum_Total_Cost,  
13 SUM(quantity_produced) AS Sum_Quantity_Produced,  
14 SUM(cost_labor+cost_material+ (machine_hours*rate_per_hour)+ cost_overhead ) / SUM(quantity_produced) AS Unit_Cost  
15  
16 FROM  
17 w_job_f  
18 JOIN w_location_d ON w_location_d.location_id = w_job_f.location_id  
19 JOIN w_time_d ON w_time_d.time_id = w_job_f.contract_date  
20 JOIN w_sub_job_f ON w_sub_job_f.job_id = w_job_f.job_id  
21 JOIN w_machine_type_d ON w_machine_type_d.machine_type_id = w_sub_job_f.machine_type_id  
22  
23 GROUP BY  
24 w_job_f.job_id,  
25 w_job_f.location_id,  
26 location_name,  
27 time_year,  
28 time_month  
29 )  
30 ORDER BY
```

Data Output

job_id	location_id	location_name	year_contract_date	month_contract_date	sum_labor_cost	sum_material_cost	sum_machine_cost	sum_overhead_cost	sum_total_cost	sum_quantity_produced	unit_cost
1	339993	1 New York	2014	4	13209.40	1083.30	8376.0000	2947.00	30045.70	48000	0.6259448033333333
2	339994	9 Montreal	2013	7	87402.08	29462.51	6660.0000	46303.58	223462.1700	728000	0.3039202344275842099
3	339995	6 Los Angeles	2015	1	84796.54	38462.71	54678.0000	36971.14	214088.3900	707100	0.3039202344275842099
4	339996	9 Montreal	2013	10	138356.56	17089.18	32760.0000	6645.79	194881.5300	310300	0.6274553322813728650
5	339997	12 Birmingham	2014	12	140706.72	35062.61	64480.0000	27312.69	287562.0200	467000	0.615744967880856531
6	339998	5 Denver	2013	7	103287.18	19235.92	14100.0000	10316.94	146960.0400	477200	0.3079632020173512154
7	339999	7 Seattle	2014	9	20414.90	13648.96	24800.0000	33996.04	87658.8000	313300	0.28117006064749441450
8	340000	6 Los Angeles	2013	7	393462.79	130794.36	239466.0000	93514.34	823377.4900	1409000	0.5850542586997901263
9	340001	11 London	2014	4	238184.10	30863.28	83796.0000	49240.56	402045.9400	1377100	0.2919511882310632511
10	340002	4 Dallas	2013	5	152994.11	38069.87	95174.0000				

Total rows: 1000 of 2569 Query complete 00:00:00.113

Successfully run. Total query runtime: 113 msec. 2569 rows affected.

BQ4:**Returns by location and sales class (quality control)**

SELECT

w_invoiceline_f.location_id,
location_name,
w_invoiceline_f.sales_class_id,
sales_class_desc,
w_time_d.time_year AS Year_Invoice_Sent_Date,
w_time_d.time_month AS Month_Invoice_Sent_Date,
SUM(quantity_shipped - invoice_quantity) AS Sum_Quantity_Returned,
SUM(quantity_shipped - invoice_quantity) * SUM(invoice_amount/invoice_quantity) AS
Sum_Amt_Return

FROM

w_invoiceline_f,
w_location_d,
w_sales_class_d,
w_time_d

WHERE

w_invoiceline_f.location_id = w_location_d.location_id
AND w_sales_class_d.sales_class_id = w_invoiceline_f.sales_class_id
AND w_time_d.time_id = w_invoiceline_f.invoice_sent_date
AND quantity_shipped > invoice_quantity

GROUP BY

w_invoiceline_f.location_id,
location_name,
w_invoiceline_f.sales_class_id,
sales_class_desc,
w_time_d.time_year,
w_time_d.time_month

ORDER BY
location_id

pgAdmin 4

Browser

JobDB/postgres@PostgreSQL 15

Query

```
1 SELECT
2   w_invoice_line_f.location_id,
3   location_name,
4   w_invoice_line_f.sales_class_id,
5   sales_class_desc,
6   w_time_d.time_year AS Year_Invoice_Sent_Date,
7   w_time_d.time_month AS Month_Invoice_Sent_Date,
8   SUM(quantity_shipped - invoice_quantity) AS Sum_Quantity_Returned,
9   SUM(quantity_shipped - invoice_quantity) * SUM(invoice_amount/invoice_quantity) AS Sum_Amt_Return
10 FROM
11   w_invoice_line_f,
12   w_location_d,
13   w_sales_class_d,
14   w_time_d
15 WHERE
16   w_invoice_line_f.location_id = w_location_d.location_id
17   AND w_sales_class_d.sales_class_id = w_invoice_line_f.sales_class_id
18   AND w_time_d.time_id = w_invoice_line_f.invoice_sent_date
19   AND quantity_shipped > invoice_quantity
20 GROUP BY
21   w_invoice_line_f.location_id,
22   location_name,
23   w_invoice_line_f.sales_class_id,
24   sales_class_desc,
25   w_time_d.time_year,
26   w_time_d.time_month
27 ORDER BY
28   location_id
```

Data Output

location_id	location_name	sales_class_id	sales_class_desc	year_invoice_sent_date	month_invoice_sent_date	sum_quantity_returned	sum_amt_return
1	New York	1	Debit Smart	2013	7	49154	193680.2838036272643060
2	New York	1	Debit Smart	2013	9	23331	44299.36788607895889
3	New York	1	Debit Smart	2013	10	32487	121074.23611111111126793
4	New York	1	Debit Smart	2013	11	22662	126295.716491927264812
5	New York	1	Debit Smart	2013	12	16350	32482.00000000000005430
6	New York	1	Debit Smart	2014	1	49948	171787.0094382022460308
7	New York	1	Debit Smart	2014	2	143338	2613256.1860026554494640
8	New York	1	Debit Smart	2014	3	18009	72754.236181203393370
9	New York	1	Debit Smart	2014	6	126616	1267024.2057079791811644
10	New York	1	Debit Smart	2014	6	83316	302399.4

Total rows: 1000 of 1191 Query complete 00:00:00.083

Successfully run. Total query runtime: 83 msec. 1191 rows affected.

BQ5:

WITH PairsCTE AS (

SELECT

w_job_f.job_id,
w_location_d.location_id,
w_location_d.location_name,
w_sales_class_d.sales_class_id,
w_sales_class_d.sales_class_desc,
w_job_f.date_promised,
max(actual_ship_date) as last_ship_date,
sum (actual_quantity) as after_ship_quantity_sum,
w_job_f.quantity_ordered,

(getBusDaysDiff(actual_ship_date,date_promised)) AS
Business_Days_Difference,
w_sub_job_f.job_id as sub_job_id

FROM

w_job_f,
w_sales_class_d,
w_location_d,
w_sub_job_f,
w_job_shipment_f

WHERE

w_job_f.job_id = w_sub_job_f.job_id
AND w_sales_class_d.sales_class_id = w_job_f.sales_class_id
AND w_location_d.location_id = w_job_f.location_id
AND w_sub_job_f.sub_job_id = w_job_shipment_f.sub_job_id
AND actual_ship_date > date_promised

GROUP BY

w_sub_job_f.sub_job_id,
w_job_f.job_id,
w_location_d.location_id,
w_location_d.location_name,
w_sales_class_d.sales_class_id,
w_sales_class_d.sales_class_desc,
Business_Days_Difference

)

SELECT

PairsCTE.job_id,
PairsCTE.location_id,
PairsCTE.location_name,
PairsCTE.sales_class_id,
PairsCTE.sales_class_desc,
PairsCTE.date_promised,
PairsCTE.last_ship_date,
PairsCTE.quantity_ordered,
PairsCTE.after_ship_quantity_sum,
PairsCTE.Business_Days_Difference

FROM PairsCTE;

BQ6:

WITH ShipmentDelaysCTE AS

(SELECT w_sub_job_f.job_id, min(Actual_Ship_Date) as Shipping_First_Date,
max(Actual_ship_date) as Shipping_Last_Date

FROM w_job_shipment_f, w_sub_job_f

WHERE W_SUB_JOB_F.sub_job_id = W_JOB_SHIPMENT_F.sub_job_id

GROUP BY W_SUB_JOB_F.job_id

),

DaysDifferenceCTE as (

SELECT

w_job_f.job_id,

w_time_d.time_id,

w_job_f.date_promised,

w_location_d.location_id,

w_location_d.location_name,

w_sales_class_d.sales_class_id,

w_sales_class_d.sales_class_desc,

w_job_f.date_ship_by,

ShipmentDelaysCTE.Shipping_First_Date,

(getBusDaysDiff(ShipmentDelaysCTE.Shipping_First_Date,
w_job_f.DATE_SHIP_BY)) AS Business_Days_Difference

FROM ShipmentDelaysCTE

INNER JOIN w_job_f on w_job_f.job_id = ShipmentDelaysCTE.job_id

INNER JOIN w_time_d on w_time_d.time_id = w_job_f.date_promised

INNER JOIN w_location_d ON w_job_f.location_id = w_location_d.location_id

INNER JOIN w_sales_class_d ON w_job_f.sales_class_id =
w_sales_class_d.sales_class_id

WHERE

```

w_location_d.location_id = w_job_f.location_id

AND w_sales_class_d.sales_class_id = w_job_f.sales_class_id

AND ShipmentDelaysCTE.job_id = w_job_f.job_id

AND ShipmentDelaysCTE.Shipping_First_Date > w_job_f.DATE_SHIP_BY

GROUP BY

w_job_f.job_id,

w_time_d.time_id,

w_location_d.location_id,

w_location_d.location_name,

w_sales_class_d.sales_class_id,

w_sales_class_d.sales_class_desc,

w_job_f.date_ship_by,

ShipmentDelaysCTE.Shipping_First_Date

```

)

```

SELECT job_id, location_id, location_name, sales_class_id, sales_class_desc, time_id,
date_ship_by, Shipping_First_Date, Business_Days_Difference

FROM DaysDifferenceCTE

```

The screenshot shows the pgAdmin 4 interface with a PostgreSQL query executed. The query uses CTEs to calculate the business days difference between shipping dates. The results table displays 10 rows of data.

job_id	location_id	location_name	sales_class_id	sales_class_desc	time_id	date_ship_by	shipping_first_date	business_days_difference
1	342393	7 Seattle	1	Credit Smart	20131002	20130923	20130926	3
2	340539	11 London	2	Credit Smart	20140728	20140721	20140723	2
3	340415	11 London	6	Loyalty NoSmart	20130409	20130404	20130409	3
4	341373	9 Montreal	2	Credit Smart	20130903	20130828	20130830	2
5	341806	10 Vancouver	3	Debit NoSmart	20140721	20140714	20140716	2
6	340542	10 Vancouver	3	Debit NoSmart	20160130	20160116	20160123	9
7	341871	2 Atlanta	4	Credit NoSmart	20140919	20140917	20140919	2
8	341349	6 Los Angeles	6	Loyalty NoSmart	20131231	20131225	20131227	2
9	342016	5 Denver	2	Credit Smart	20141112	20141103	20141106	3
10	340532	11 London	6	Loyalty NoSmart	20131210	20131204	20131206	2

Total rows: 120 of 120 Query complete 00:00:00.238 Successfully run. Total query runtime: 238 msec. 120 rows affected.

AQ1:

SELECT

location_name,

w_time_d.time_year AS Contract_Year,

w_time_d.Time_month AS Contract_Month,

SUM(Quantity_ordered*Unit_price) AS Sum_Job_Amount,

SUM(SUM(Quantity_ordered*Unit_price)) OVER (PARTITION BY location_name,
w_time_d.time_year ORDER BY time_month ROWS UNBOUNDED PRECEDING) AS
Cum_Amount_Ordered

FROM

w_job_f,

w_location_d,

w_sales_class_d,

w_time_d

WHERE

w_location_d.location_id = w_job_f.location_id

AND w_sales_class_d.sales_class_id = w_job_f.sales_class_id

AND w_time_d.time_id = w_job_f.contract_date

GROUP BY

location_name,

w_time_d.time_year,

w_time_d.Time_month

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Server Enterprise Explorer' tree shows the 'JobDB' database selected. The 'Tables (3)' folder is expanded, showing 'diacodetrauma', 'diacode', and 'diacode_type'. The 'Columns (6)' folder is also expanded, showing 'diacode', 'diacode_type', 'diacode_text', 'diacode_value', 'diacode_value_text', and 'diacode_value_value'. The main window shows a SQL query in the 'Query History' tab. The query is a complex SELECT statement with multiple joins and aggregations. The results are displayed in the 'Data Output' tab, showing 10 rows of data. The status bar at the bottom indicates 'Successfully run. Total query runtime: 67 msec. 313 rows affected.'

Query:

```

1 SELECT
2     location_name,
3     w_time_d.time_year AS Contract_Year,
4     w_time_d.time_month AS Contract_Month,
5     SUM(Quantity_orderedUnit_price) AS Sum_Job_Amount,
6     SUM(SUM(Quantity_orderedUnit_price)) OVER (PARTITION BY location_name, w_time_d.time_year ORDER BY time_month ROWS UNBOUNDED PRECEDING) AS Cum_Amount_Ordered
7 FROM
8     w_job_f,
9     w_location_d,
10    w_sales_class_d,
11    w_time_d
12 WHERE
13     w_location_d.location_id = w_job_f.location_id
14     AND w_sales_class_d.sales_class_id = w_job_f.sales_class_id
15     AND w_time_d.time_id = w_job_f.contract_date
16 GROUP BY
17     location_name,
18     w_time_d.time_year,
19     w_time_d.time_month
20

```

Data Output:

location_name	contract_year	contract_month	num_job_amount	cum_amount_ordered
Atlanta	2013	1	115332.00	115332.00
Atlanta	2013	2	3048906.00	4204238.00
Atlanta	2013	3	2922105.00	7126343.00
Atlanta	2013	4	4975735.00	12102098.00
Atlanta	2013	5	9651206.00	21753304.00
Atlanta	2013	6	3973459.00	25696763.00
Atlanta	2013	7	6319335.00	32226298.00
Atlanta	2013	8	5433568.00	37699866.00
Atlanta	2013	9	4412602.00	42072468.00
Atlanta	2013	10	9211690.00	51384158.00

Total rows: 313 of 313 Query complete 00:00:00.057

Successfully run. Total query runtime: 67 msec. 313 rows affected.

AQ2:

SELECT

location_name,

w_time_d.time_year AS Contract_Year,

w_time_d.Time_month AS Contract_Month,

AVG(Quantity_ordered*Unit_price) AS Average_job_amount_ordered,

AVG(SUM(Quantity_ordered*Unit_price)) OVER (PARTITION BY location_name ORDER BY
w_time_d.time_year, w_time_d.Time_month ROWS BETWEEN 1 PRECEDING AND 11
FOLLOWING) AS Moving_Avg_Avg_Amt_Ordered

FROM

w_job_f,

w_location_d,

w_sales_class_d,

w_time_d

WHERE

w_location_d.location_id = w_job_f.location_id

AND w_sales_class_d.sales_class_id = w_job_f.sales_class_id

AND w_time_d.time_id = w_job_f.contract_date

GROUP BY

location_name,

w_time_d.time_year,

w_time_d.Time_month

pgAdmin 4

File Object Tools Help

Browser

- Databases (8)
 - JobDB
 - Casts
 - Catalogs
 - Event Triggers
 - Extensions
 - Foreign Data Wrappers
 - Language
 - Publications
 - Schemas (1)
 - public
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (3)
 - diagcode
 - diagcode
 - diagcode

Query

```
1 SELECT
2   location_name,
3   w_time_d.time_year AS Contract_Year,
4   w_time_d.time_month AS Contract_Month,
5   AVG(Quantity_ordered*Unit_price) AS Average_job_amount_ordered,
6   AVG(SUM(Quantity_ordered*Unit_price)) OVER (PARTITION BY location_name ORDER BY w_time_d.time_year, w_time_d.time_month ROWS BETWEEN 1 PRECEDING AND 11 FOLLOWING) AS Moving_Avg_Amt
7 FROM
8   w_job_f,
9   w_location_d,
10  w_sales_class_d,
11  w_time_d
12 WHERE
13   w_location_d.location_id = w_job_f.location_id
14   AND w_sales_class_d.sales_class_id = w_job_f.sales_class_id
15   AND w_time_d.time_id = w_job_f.contract_date
16 GROUP BY
17   location_name,
18   w_time_d.time_year,
19   w_time_d.time_month
20
```

Data Output

	location_name	contract_year	contract_month	average_job_amount_ordered	moving_avg_amt_ordered
1	Atlanta	2013	1	1155352.000000000000	5135684.953333333333
2	Atlanta	2013	2	528151.000000000000	5045457.615384615385
3	Atlanta	2013	3	487017.500000000000	5296013.923076923077
4	Atlanta	2013	4	497573.500000000000	5407689.461538461538
5	Atlanta	2013	5	491560.300000000000	5305629.623076923077
6	Atlanta	2013	6	567665.971428571429	5472910.692307692308
7	Atlanta	2013	7	526411.250000000000	4961423.000000000000
8	Atlanta	2013	8	603729.777777777778	5168714.746230769231
9	Atlanta	2013	9	735433.666666666667	4994871.615384615385
10	Atlanta	2013	10	716293.846153846154	5007976.461538461538

Total rows: 313 of 313 Query complete 00:00:00.135

Successfully run. Total query runtime: 135 msec. 313 rows affected.

AQ3:

WITH Inv_Rev_SummaryCTE AS (

SELECT w_job_f.job_id, w_job_f.location_id, location_name, unit_price, quantity_ordered,
w_time_d.time_year, w_time_d.time_month,

SUM(w_invoiceline_f.invoice_amount) AS Sum_Invoice_Amt,
SUM(w_invoiceline_f.invoice_quantity) AS Sum_Invoice_Quantity

FROM w_job_f, w_location_d, w_time_d, w_invoiceline_f

WHERE w_job_f.location_id = w_location_d.location_id

AND w_job_f.contract_date = w_time_d.time_id

AND w_job_f.location_id = w_invoiceline_f.location_id

GROUP BY w_job_f.job_id, w_job_f.location_id, location_name, unit_price, quantity_ordered,
w_time_d.time_year, w_time_d.time_month

), Loc_Subjob_SummaryCTE AS (

SELECT w_job_f.job_id, w_job_f.location_id, location_name, w_time_d.time_year,
w_time_d.time_month,

SUM(cost_labor) AS total_Labor_cost, SUM(cost_material) AS Total_material_cost,

SUM(machine_hours * rate_per_hour) AS total_machine_cost, SUM(cost_overhead) AS
Total_overhead_cost,

SUM(cost_labor + cost_material + (machine_hours * rate_per_hour) + cost_overhead) AS
Total_Cost,

SUM(quantity_produced) AS SumQuantityProduced,

SUM(cost_labor + cost_material + (machine_hours * rate_per_hour) + cost_overhead) /
SUM(quantity_produced) AS Unit_Cost

FROM w_job_f

JOIN w_location_d ON w_location_d.location_id = w_job_f.location_id

JOIN w_time_d ON w_time_d.time_id = w_job_f.contract_date

JOIN w_sub_job_f ON w_sub_job_f.job_id = w_job_f.job_id

JOIN w_machine_type_d ON w_machine_type_d.machine_type_id =
w_sub_job_f.machine_type_id

GROUP BY w_job_f.job_id, w_job_f.location_id, location_name, time_year, time_month

ORDER BY job_id ASC

)

SELECT Inv_Rev_SummaryCTE.location_name, Inv_Rev_SummaryCTE.time_year AS
Contract_Year,

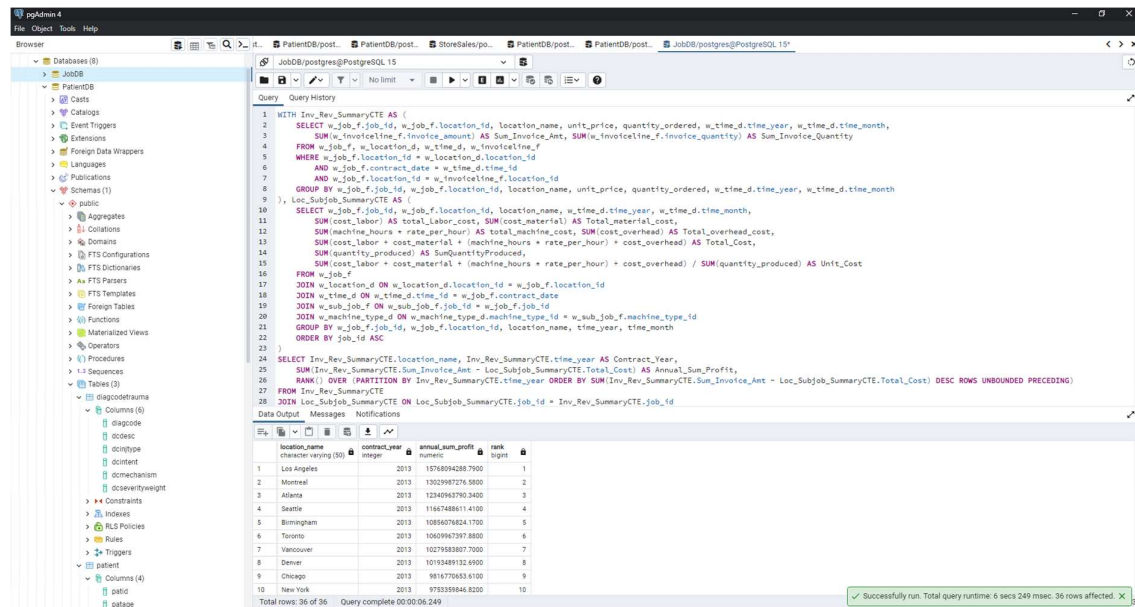
SUM(Inv_Rev_SummaryCTE.Sum_Invoice_Amt - Loc_Subjob_SummaryCTE.Total_Cost) AS
Annual_Sum_Profit,

RANK() OVER (PARTITION BY Inv_Rev_SummaryCTE.time_year ORDER BY
SUM(Inv_Rev_SummaryCTE.Sum_Invoice_Amt - Loc_Subjob_SummaryCTE.Total_Cost) DESC
ROWS UNBOUNDED PRECEDING)

FROM Inv_Rev_SummaryCTE

JOIN Loc_Subjob_SummaryCTE ON Loc_Subjob_SummaryCTE.job_id =
Inv_Rev_SummaryCTE.job_id

GROUP BY Inv_Rev_SummaryCTE.location_name, Inv_Rev_SummaryCTE.time_year



The screenshot shows a PostgreSQL query editor with a complex SQL query and its results. The query is as follows:

```

1 WITH Inv_Rev_SummaryCTE AS (
2   SELECT w_job.f.job_id, w_job.f.location_id, location_name, unit_price, quantity_ordered, w_time.d.time_year, w_time.d.time_month,
3   SUM(w_invoice.f.invoice_amount) AS Sum_Invoice_Amt, SUM(w_invoice.f.invoice_quantity) AS Sum_Invoice_Quantity
4   FROM w_job.f, w_location.d, w_time.d, w_invoice.f
5   WHERE w_job.f.location_id = w_location.d.location_id
6   AND w_job.f.contract_date = w_time.d.time_id
7   AND w_job.f.location_id = w_invoice.f.location_id
8   GROUP BY w_job.f.job_id, w_job.f.location_id, location_name, unit_price, quantity_ordered, w_time.d.time_year, w_time.d.time_month
9 ), Loc_Subjob_SummaryCTE AS (
10  SELECT w_job.f.job_id, w_job.f.location_id, location_name, w_time.d.time_year, w_time.d.time_month,
11  SUM(cost_labor) AS total_labor_cost, SUM(cost_material) AS Total_material_cost,
12  SUM(machine_hours * rate_per_hour) AS total_machine_cost, SUM(cost_overhead) AS Total_overhead_cost,
13  SUM(cost_labor + cost_material + (machine_hours * rate_per_hour) + cost_overhead) AS Total_Cost,
14  SUM(quantity_produced) AS SumQuantityProduced,
15  SUM(cost_labor + cost_material + (machine_hours * rate_per_hour) + cost_overhead) / SUM(quantity_produced) AS Unit_Cost
16  FROM w_job.f
17  JOIN w_location.d ON w_location.d.location_id = w_job.f.location_id
18  JOIN w_time.d ON w_time.d.time_id = w_job.f.contract_date
19  JOIN w_sub_job.f ON w_sub_job.f.job_id = w_job.f.job_id
20  JOIN w_machine_type.d ON w_machine_type.d.machine_type_id = w_sub_job.f.machine_type_id
21  GROUP BY w_job.f.job_id, w_job.f.location_id, location_name, time_year, time_month
22  ORDER BY job_id ASC
23 )
24 SELECT Inv_Rev_SummaryCTE.location_name, Inv_Rev_SummaryCTE.time_year AS Contract_Year,
25 SUM(Inv_Rev_SummaryCTE.Sum_Invoice_Amt - Loc_Subjob_SummaryCTE.Total_Cost) AS Annual_Sum_Profit,
26 RANK() OVER (PARTITION BY Inv_Rev_SummaryCTE.time_year ORDER BY SUM(Inv_Rev_SummaryCTE.Sum_Invoice_Amt - Loc_Subjob_SummaryCTE.Total_Cost) DESC ROWS UNBOUNDED PRECEDING)
27 FROM Inv_Rev_SummaryCTE
28 JOIN Loc_Subjob_SummaryCTE ON Loc_Subjob_SummaryCTE.job_id = Inv_Rev_SummaryCTE.job_id

```

The results table shows the following data:

location_name	contract_year	annual_sum_profit	rank
Los Angeles	2013	1576809428.7900	1
Montreal	2013	13029987276.5800	2
Atlanta	2013	12340963790.3400	3
Seattle	2013	11667488611.4100	4
Birmingham	2013	10856076824.1700	5
Tampa	2013	10609697997.8900	6
Vancouver	2013	10279583807.7000	7
Denver	2013	10193489132.8900	8
Chicago	2013	9816770653.6100	9
New York	2013	9783398648.8200	10

Total rows: 36 of 36 Query complete 00:00:06.249

Successfully run. Total query runtime: 6 secs 249 msec. 36 rows affected.

AQ4:

WITH Inv_Rev_SummaryCTE AS (

SELECT

w_job_f.job_id,

w_job_f.location_id,

location_name,

unit_price,

quantity_ordered,

w_time_d.time_year,

w_time_d.time_month,

SUM(w_invoiceline_f.invoice_amount) AS Sum_Invoice_Amt,

SUM(w_invoiceline_f.invoice_quantity)

FROM w_job_f, w_location_d, w_time_d, w_invoiceline_f

WHERE w_job_f.location_id = w_location_d.location_id

AND w_job_f.contract_date = w_time_d.time_id

AND w_job_f.location_id = w_invoiceline_f.location_id

GROUP BY

w_job_f.job_id,

w_job_f.location_id,

location_name,

unit_price,

quantity_ordered,

w_time_d.time_year,

w_time_d.time_month

),

Loc_Subjob_SummaryCTE AS (

SELECT

w_job_f.job_id,

w_job_f.location_id,

```

        location_name,
        w_time_d.time_year,
        w_time_d.time_month,
        SUM(cost_labor) AS total_Labor_cost,
        SUM(cost_material) AS Total_material_cost,
        SUM(machine_hours*rate_per_hour) AS total_machine_cost,
        SUM(cost_overhead) AS Total_overhead_cost,
        SUM(cost_labor+ cost_material+ (machine_hours*rate_per_hour)+ cost_overhead ) AS
Total_Cost,
        SUM(quantity_produced) AS SumQuantityProduced,
        SUM(cost_labor+ cost_material+ (machine_hours*rate_per_hour)+ cost_overhead )/
SUM(quantity_produced) AS Unit_Cost
FROM w_job_f
JOIN w_location_d ON w_location_d.location_id = w_job_f.location_id
JOIN w_time_d ON w_time_d.time_id = w_job_f.contract_date
JOIN w_sub_job_f ON w_sub_job_f.job_id = w_job_f.job_id
JOIN w_machine_type_d ON w_machine_type_d.machine_type_id =
w_sub_job_f.machine_type_id
GROUP BY
        w_job_f.job_id,
        w_job_f.location_id,
        location_name,
        time_year,
        time_month
ORDER BY job_id ASC
)
SELECT
        Inv_Rev_SummaryCTE.location_name,
        Inv_Rev_SummaryCTE.time_year AS Contract_Year,
        SUM(Inv_Rev_SummaryCTE.Sum_Invoice_Amt-Loc_Subjob_SummaryCTE.Total_Cost) AS
Annual_Sum_Profit,

```

(SUM((Inv_Rev_SummaryCTE.Sum_Invoice_Amt-Loc_Subjob_SummaryCTE.Total_Cost)/
Sum_Invoice_Amt)) AS Annual_Profit_Margin,

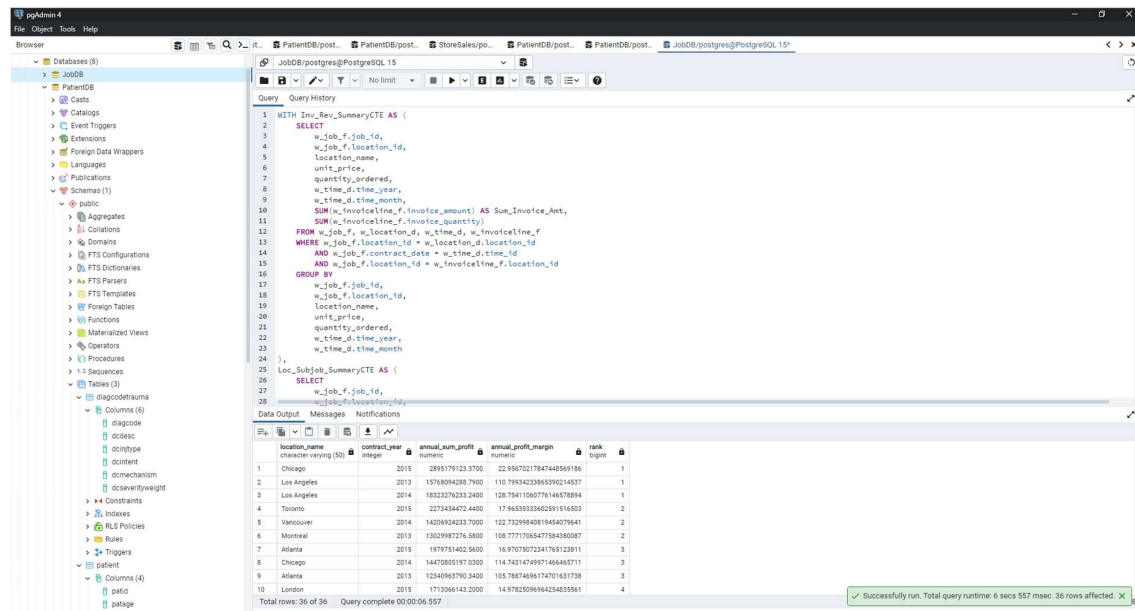
RANK() OVER (PARTITION BY Inv_Rev_SummaryCTE.time_year ORDER BY
(SUM((Inv_Rev_SummaryCTE.Sum_Invoice_Amt-
Loc_Subjob_SummaryCTE.Total_Cost)/Sum_Invoice_Amt)) DESC ROWS UNBOUNDED
PRECEDING)

FROM Inv_Rev_SummaryCTE

JOIN Loc_Subjob_SummaryCTE ON Loc_Subjob_SummaryCTE.job_id =
Inv_Rev_SummaryCTE.job_id

GROUP BY Inv_Rev_SummaryCTE.location_name, Inv_Rev_SummaryCTE.time_year

ORDER BY RANK;



The screenshot shows the pgAdmin 4 interface with a SQL query executed in the 'Query' tab. The query is a complex SQL statement involving CTEs and window functions. The 'Data Output' tab shows the results of the query, which is a table with 10 rows and 5 columns: location_name, contract_year, annual_sum_profit, annual_profit_margin, and rank. The results are sorted by rank in descending order.

Query:

```

1 WITH Inv_Rev_SummaryCTE AS (
2     SELECT
3         w_job_f.job_id,
4         w_job_f.location_id,
5         location_name,
6         unit_price,
7         quantity_ordered,
8         w_time_d.time_year,
9         w_time_d.time_month,
10        SUM(w_invoice_line_f.invoice_amount) AS Sum_Invoice_Amt,
11        SUM(w_invoice_line_f.invoice_quantity)
12    FROM w_job_f, w_location_d, w_time_d, w_invoice_line_f
13    WHERE w_job_f.location_id = w_location_d.location_id
14        AND w_job_f.contract_date = w_time_d.time_id
15        AND w_job_f.location_id = w_invoice_line_f.location_id
16    GROUP BY
17        w_job_f.job_id,
18        w_job_f.location_id,
19        location_name,
20        unit_price,
21        quantity_ordered,
22        w_time_d.time_year,
23        w_time_d.time_month
24 ),
25 Loc_Subjob_SummaryCTE AS (
26     SELECT
27         w_job_f.job_id,
28         w_job_f.location_id,
29         w_job_f.contract_date,
30         w_job_f.location_id
31    FROM w_job_f
32 )
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Data Output:

location_name	contract_year	annual_sum_profit	annual_profit_margin	rank
Chicago	2015	2895179123.3700	22.95670217847448569186	1
Los Angeles	2013	1076809428.7900	110.793423360390214537	1
Los Angeles	2014	1923276233.2400	128.78411060776146578984	1
Toronto	2015	2273434472.4400	17.963933360291516503	2
Vancouver	2014	14226924233.7000	122.73299840819454079641	2
Montreal	2013	13029987276.8800	108.77717065477584880887	2
Atlanta	2015	1979751402.5600	16.9707507234176123811	3
Chicago	2014	1447020197.0300	114.7614749971466460711	3
Atlanta	2013	12340983790.3400	105.788746616701631738	3
London	2015	1713066143.2000	14.9782059694254835561	4

Total rows: 36 of 36 Query complete 00:00:06.557

Successfully run. Total query runtime: 6 secs 557 msec. 36 rows affected.

AQ5:

```
WITH Inv_Rev_SummaryCTE AS (select w_job_f.job_id, w_job_f.location_id, location_name,  
unit_price, quantity_ordered, w_time_d.time_year, w_time_d.time_month,  
sum(w_invoiceline_f.invoice_amount) as Sum_Invoice_Amt,  
sum(w_invoiceline_f.invoice_quantity)
```

```
from w_job_f, w_location_d, w_time_d, w_invoiceline_f
```

```
where w_job_f.location_id = w_location_d.location_id
```

```
and w_job_f.contract_date = w_time_d.time_id
```

```
and w_job_f.location_id = w_invoiceline_f.location_id
```

```
group by w_job_f.job_id, w_job_f.location_id, location_name, unit_price, quantity_ordered,  
w_time_d.time_year, w_time_d.time_month)
```

```
,
```

```
Loc_Subjob_SummaryCTE as (
```

```
select w_job_f.job_id, w_job_f.location_id, location_name, w_time_d.time_year,  
w_time_d.time_month, sum(cost_labor) as total_Labor_cost,
```

```
sum(cost_material) as Total_material_cost, sum(machine_hours*rate_per_hour) as  
total_machine_cost, sum(cost_overhead) as Total_overhead_cost,
```

```
sum (cost_labor+ cost_material+ (machine_hours*rate_per_hour)+ cost_overhead )as  
Total_Cost,
```

```
sum(quantity_produced) as SumQuantityProduced,
```

```
sum(cost_labor+ cost_material+ (machine_hours*rate_per_hour)+ cost_overhead )/  
sum(quantity_produced) as Unit_Cost
```

```
FROM w_job_f
```

```
join w_location_d on w_location_d.location_id = w_job_f.location_id
```

```
join w_time_d on w_time_d.time_id = w_job_f.contract_date
```

```
join w_sub_job_f on w_sub_job_f.job_id = w_job_f.job_id
```

```
join w_machine_type_d on w_machine_type_d.machine_type_id =  
w_sub_job_f.machine_type_id
```

```
group by w_job_f.job_id, w_job_f.location_id, location_name, time_year, time_month
```

```
order by job_id asc
```


)

select Inv_Rev_SummaryCTE.job_id, Inv_Rev_SummaryCTE.location_name,
Inv_Rev_SummaryCTE.time_year as Contract_Year, Inv_Rev_SummaryCTE.time_month as
Contract_Month,

sum((Inv_Rev_SummaryCTE.Sum_Invoice_Amt-Loc_Subjob_SummaryCTE.Total_Cost)/
Sum_Invoice_Amt) as Profit_Margin,

Percent_rank() over (order by sum((Inv_Rev_SummaryCTE.Sum_Invoice_Amt-
Loc_Subjob_SummaryCTE.Total_Cost)/Sum_Invoice_Amt)) as PercentRank

from Inv_Rev_SummaryCTE

join Loc_Subjob_SummaryCTE on Loc_Subjob_SummaryCTE.job_id =
Inv_Rev_SummaryCTE.job_id

group by Inv_Rev_SummaryCTE.job_id, Inv_Rev_SummaryCTE.location_name,
Inv_Rev_SummaryCTE.time_year, Inv_Rev_SummaryCTE.time_month

The screenshot displays the pgAdmin 4 interface. On the left, the 'Databases (8)' tree shows the 'public' schema. The main pane shows a SQL query with CTEs for 'Inv_Rev_SummaryCTE' and 'Loc_Subjob_SummaryCTE'. The query calculates profit margin and percent rank. The 'Data Output' pane at the bottom shows the results of the query, which includes columns for job_id, location_name, contract_year, contract_month, profit_margin, and percentrank. The results are sorted by job_id.

job_id	location_name	contract_year	contract_month	profit_margin	percentrank
341491	Dallas	2014	6	0.9914889219008321605	0
342070	Dallas	2013	6	0.9918232424210022505	0.0003894080996884735
341347	Dallas	2013	3	0.9918997480196666145	0.000778816199376847
340701	New York	2014	12	0.99239676923611072353	0.0011682424990564205
341183	Vancouver	2014	2	0.99258419569774295948	0.001557632298753894
340281	New York	2014	10	0.9929378925643741081	0.001947640498423576
341559	Dallas	2014	1	0.99291616010076466665	0.002326449599138841
341013	Dallas	2014	6	0.993050819328748864	0.002728566678193145
342390	Vancouver	2014	2	0.993074230172882893	0.003115264797507788
342282	Atlanta	2015	2	0.9930510151240548784	0.0035046728971962616

Total rows: 1000 of 2569 Query complete 00:00:05.907

Successfully run. Total query runtime: 5 secs 907 msec. 2569 rows affected.

AQ6:

```
WITH Inv_Rev_SummaryCTE AS (select w_job_f.job_id, w_job_f.location_id, location_name,  
unit_price, quantity_ordered, w_time_d.time_year, w_time_d.time_month,  
sum(w_invoiceline_f.invoice_amount) as Sum_Invoice_Amt,  
sum(w_invoiceline_f.invoice_quantity)
```

```
from w_job_f, w_location_d, w_time_d, w_invoiceline_f
```

```
where w_job_f.location_id = w_location_d.location_id
```

```
and w_job_f.contract_date = w_time_d.time_id
```

```
and w_job_f.location_id = w_invoiceline_f.location_id
```

```
group by w_job_f.job_id, w_job_f.location_id, location_name, unit_price, quantity_ordered,  
w_time_d.time_year, w_time_d.time_month)
```

```
,
```

```
Loc_Subjob_SummaryCTE as (
```

```
select w_job_f.job_id, w_job_f.location_id, location_name, w_time_d.time_year,  
w_time_d.time_month, sum(cost_labor) as total_Labor_cost,
```

```
sum(cost_material) as Total_material_cost, sum(machine_hours*rate_per_hour) as  
total_machine_cost, sum(cost_overhead) as Total_overhead_cost,
```

```
sum (cost_labor+ cost_material+ (machine_hours*rate_per_hour)+ cost_overhead )as  
Total_Cost,
```

```
sum(quantity_produced) as SumQuantityProduced,
```

```
sum(cost_labor+ cost_material+ (machine_hours*rate_per_hour)+ cost_overhead )/  
sum(quantity_produced) as Unit_Cost
```

```
FROM w_job_f
```

```
join w_location_d on w_location_d.location_id = w_job_f.location_id
```

```
join w_time_d on w_time_d.time_id = w_job_f.contract_date
```

```
join w_sub_job_f on w_sub_job_f.job_id = w_job_f.job_id
```

```
join w_machine_type_d on w_machine_type_d.machine_type_id =  
w_sub_job_f.machine_type_id
```

```
group by w_job_f.job_id, w_job_f.location_id, location_name, time_year, time_month
```

```
order by job_id asc
```

)

select job_id, location_name, time_year as Contract_Year, time_month as Contract_Month,
Annual_Profit_Margin, PercentRankProfit

from (select Inv_Rev_SummaryCTE.job_id, Inv_Rev_SummaryCTE.location_name,
Inv_Rev_SummaryCTE.time_year, Inv_Rev_SummaryCTE.time_month,

sum((Inv_Rev_SummaryCTE.Sum_Invoice_Amt-Loc_Subjob_SummaryCTE.Total_Cost)/
Sum_Invoice_Amt) as Annual_Profit_Margin,

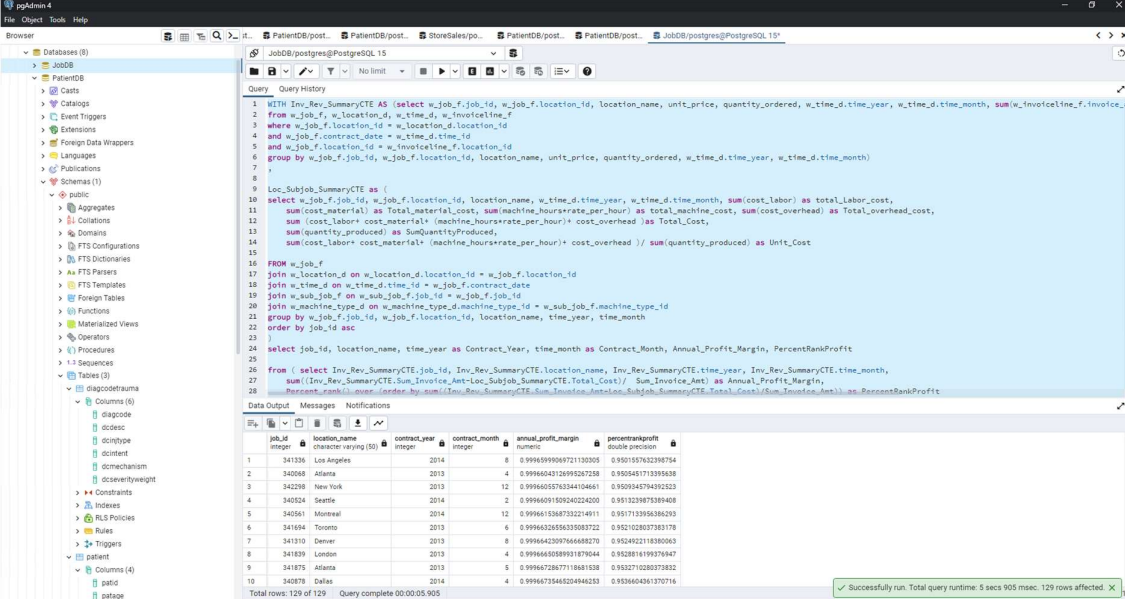
Percent_rank() over (order by sum((Inv_Rev_SummaryCTE.Sum_Invoice_Amt-
Loc_Subjob_SummaryCTE.Total_Cost)/Sum_Invoice_Amt)) as PercentRankProfit

from Inv_Rev_SummaryCTE

join Loc_Subjob_SummaryCTE on Loc_Subjob_SummaryCTE.job_id =
Inv_Rev_SummaryCTE.job_id

group by Inv_Rev_SummaryCTE.job_id, Inv_Rev_SummaryCTE.location_name,
Inv_Rev_SummaryCTE.time_year, Inv_Rev_SummaryCTE.time_month) X

where PercentRankProfit > 0.95



Query:

```
WITH Inv_Rev_SummaryCTE AS (select w_job.f.job_id, w_job.f.location_id, location_name, unit_price, quantity_ordered, w_time.d.time_year, w_time.d.time_month, sum(w_invoice.f.invoice_d
from w_job.f, w_location_d, w_time_d, w_invoice.f
where w_job.f.location_id = w_location_d.location_id
and w_job.f.contract_date = w_time_d.time_id
and w_job.f.location_id = w_invoice.f.location_id
group by w_job.f.job_id, w_job.f.location_id, location_name, unit_price, quantity_ordered, w_time.d.time_year, w_time.d.time_month)
, Loc_Subjob_SummaryCTE AS (
select w_job.f.job_id, w_job.f.location_id, location_name, w_time.d.time_year, w_time.d.time_month, sum(cost_labor) as total_labor_cost,
sum(cost_material) as total_material_cost, sum(machine_hours*rate_per_hour) as total_machine_cost, sum(cost_overhead) as total_overhead_cost,
sum(cost_labor+cost_material+ (machine_hours*rate_per_hour)+ cost_overhead) as Total_Cost,
sum(quantity_produced) as SumQuantityProduced,
sum(cost_labor+ cost_material+ (machine_hours*rate_per_hour)+ cost_overhead )/ sum(quantity_produced) as Unit_Cost
FROM w_job.f
join w_location_d on w_location_d.location_id = w_job.f.location_id
join w_time_d on w_time_d.time_id = w_job.f.contract_date
join w_sub_job_f on w_sub_job_f.job_id = w_job.f.job_id
join w_machine_type_d on w_machine_type_d.machine_type_id = w_sub_job_f.machine_type_id
group by w_job.f.job_id, w_job.f.location_id, location_name, time_year, time_month
order by job_id asc)
select job_id, location_name, time_year as Contract_Year, time_month as Contract_Month, Annual_Profit_Margin, PercentRankProfit
from ( select Inv_Rev_SummaryCTE.job_id, Inv_Rev_SummaryCTE.location_name, Inv_Rev_SummaryCTE.time_year, Inv_Rev_SummaryCTE.time_month,
sum((Inv_Rev_SummaryCTE.Sum_Invoice_Amt-Loc_Subjob_SummaryCTE.Total_Cost)/ Sum_Invoice_Amt) as Annual_Profit_Margin,
Percent_rank() over (order by sum((Inv_Rev_SummaryCTE.Sum_Invoice_Amt-Loc_Subjob_SummaryCTE.Total_Cost)/Sum_Invoice_Amt)) as PercentRankProfit
```

Data Output:

job_id	location_name	contract_year	contract_month	annual_profit_margin	percentrankprofit
1	341326 Los Angeles	2014	6	0.9994999506721130305	0.9501557632398754
2	340266 Atlanta	2013	4	0.99946605712695037268	0.9505481713398588
3	342298 New York	2013	12	0.9994605576344104461	0.950548764935323
4	340524 Seattle	2014	2	0.9994605150924024005	0.9513239875389408
5	340561 Montreal	2014	12	0.99946153687332214911	0.9517133956386293
6	341694 Toronto	2013	6	0.999462656635083722	0.9521028037385178
7	341310 Denver	2013	8	0.9994642309766668270	0.9524922118380063
8	341259 London	2013	4	0.999466509919787044	0.9528161957931467
9	341875 Atlanta	2013	5	0.9994672671118681558	0.9532710280373852
10	340870 Dallas	2014	4	0.9994675465204946253	0.9536043613707716

Total rows: 129 of 129 Query complete 00:00:05.905

Successfully run. Total query runtime: 5 secs 905 msec. 129 rows affected.

AQ7:

```
select sales_class_desc, w_time_d.time_year as Year_Date_Sent,  
       SUM(quantity_shipped - invoice_quantity) as Sum_Return_Quantity,
```

```
       rank() over (partition by w_time_d.time_year order by SUM(quantity_shipped -  
invoice_quantity) desc)
```

```
from w_invoiceline_f, w_location_d, w_sales_class_d, w_time_d
```

```
where w_invoiceline_f.location_id = w_location_d.location_id
```

```
and w_sales_class_d.sales_class_id = w_invoiceline_f.sales_class_id
```

```
and w_time_d.time_id = w_invoiceline_f.invoice_sent_date
```

```
and quantity_shipped > invoice_quantity
```

```
group by sales_class_desc, w_time_d.time_year
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Databases (1)' tree is expanded to show 'PatientDB'. The 'Query' tab is active, displaying the following SQL query:

```
1 select sales_class_desc, w_time_d.time_year as Year_Date_Sent,  
2 SUM(quantity_shipped - invoice_quantity) as Sum_Return_Quantity,  
3  
4 rank() over (partition by w_time_d.time_year order by SUM(quantity_shipped - invoice_quantity) desc)  
5  
6 from w_invoiceline_f, w_location_d, w_sales_class_d, w_time_d  
7 where w_invoiceline_f.location_id = w_location_d.location_id  
8 and w_sales_class_d.sales_class_id = w_invoiceline_f.sales_class_id  
9 and w_time_d.time_id = w_invoiceline_f.invoice_sent_date  
10 and quantity_shipped > invoice_quantity  
11 group by sales_class_desc, w_time_d.time_year
```

The 'Data Output' tab shows the results of the query. The table has 4 columns: sales_class_desc, year_date_sent, sum_return_quantity, and rank. The results are as follows:

sales_class_desc	year_date_sent	sum_return_quantity	rank
Prepaid NotSmart	2013	36248996	1
Debit NotSmart	2013	20914124	2
Lowly NotSmart	2013	29671632	3
Debit Smart	2013	29395008	4
Credit NotSmart	2013	27088040	5
Credit Smart	2013	25002144	6
Debit Smart	2014	52273740	1
Credit Smart	2014	49795094	2
Lowly NotSmart	2014	49544320	3
Credit NotSmart	2014	43540580	4

The status bar at the bottom indicates: 'Total rows: 18 of 18 Query complete 00:00:00.076 Successfully run. Total query runtime: 76 msec. 18 rows affected.'

AQ8:

SELECT

```
    sales_class_desc,  
  
    w_time_d.time_year,  
  
    SUM(quantity_shipped - invoice_quantity) as Sum_Return_Quantity,  
  
    sum((quantity_shipped - invoice_quantity)*(invoice_amount/invoice_quantity)) as  
Sum_Amount_Return,  
  
    SUM(quantity_shipped - invoice_quantity)/sum(SUM(quantity_shipped - invoice_quantity))  
over (partition by w_time_d.time_year) as RTP_Sum_Return_Quantity
```

FROM

```
    w_invoiceline_f,  
  
    w_location_d,  
  
    w_sales_class_d,  
  
    w_time_d
```

WHERE

```
    w_invoiceline_f.location_id = w_invoiceline_f.location_id  
  
    and w_sales_class_d.sales_class_id = w_invoiceline_f.sales_class_id  
  
    and w_time_d.time_id = w_invoiceline_f.invoice_sent_date  
  
    and quantity_shipped > invoice_quantity
```

GROUP BY

```
    sales_class_desc,  
  
    w_time_d.time_year
```

pgAdmin 4

File Object Tools Help

Browser

- Databases (8)
 - PostgreSQL
 - Public
 - Aggregates
 - diagnosedtrauma
 - Columns (6)
 - diagnosedtrauma
 - diagnosedtrauma
 - diagnosedtrauma
 - diagnosedtrauma
 - diagnosedtrauma
 - diagnosedtrauma

Query

```
1 SELECT
2   sales_class_desc,
3   w_time_d.time_year,
4   SUM(quantity_shipped - invoice_quantity) as Sum_Return_Quantity,
5   sum((quantity_shipped - invoice_quantity)*(invoice_amount/invoice_quantity)) as Sum_Amount_Return,
6   SUM(quantity_shipped - invoice_quantity)/sum(SUM(quantity_shipped - invoice_quantity)) over (partition by w_time_d.time_year) as RTP_Sum_Return_Quantity
7 FROM
8   w_invoice_line_f,
9   w_location_d,
10  w_sales_class_d,
11  w_time_d
12 WHERE
13   w_invoice_line_f.location_id = w_invoice_line_f.location_id
14   and w_sales_class_d.sales_class_id = w_invoice_line_f.sales_class_id
15   and w_time_d.time_id = w_invoice_line_f.invoice_sent_date
16   and quantity_shipped > invoice_quantity
17 GROUP BY
18   sales_class_desc,
19   w_time_d.time_year
20
```

Data Output

sales_class_desc	time_year	sum_return_quantity	sum_amount_return	rtp_sum_return_quantity
character varying (255)	integer	bigint	numeric	numeric
1 Debt Smart	2013	2995608	57013582.95216454023192814086	0.16482549935644917646
2 Prepaid NoSmart	2013	2625596	34595287.1044218556414740212	0.200105059601962628
3 Credit NoSmart	2013	2700840	28913618.64532596747543317136	0.1512577773456614186
4 Debt NoSmart	2013	30914124	29448727.5856788695166655952	0.17813323481456695102
5 Credit Smart	2013	25002144	47992328.6871400708298248	0.14002344261864305801
6 Loyalty NoSmart	2013	29971632	29013319.1825840723267273604	0.16785484850875324635
7 Loyalty NoSmart	2014	4952420	4741056.8881893563124110114	0.1772614918984246861
8 Credit NoSmart	2014	4354580	42047817.1238957697439651308	0.1550968272086809546
9 Credit Smart	2014	4978384	9584781.4967648619734997712	0.17815993926183200469
10 Debt NoSmart	2014	42288808	40787549.2061228999184610276	0.1512650832975334680

Total rows: 18 of 18 Query complete 00:00:00.126

Successfully run. Total query runtime: 126 msec. 18 rows affected.

AQ9:

WITH ShipmentDelaysCTE AS

(SELECT w_sub_job_f.job_id, min(Actual_Ship_Date) as Shipping_First_Date,
max(Actual_ship_date) as Shipping_Last_Date

FROM w_job_shipment_f, w_sub_job_f

WHERE W_SUB_JOB_F.sub_job_id = W_JOB_SHIPMENT_F.sub_job_id

GROUP BY W_SUB_JOB_F.job_id

),

DaysDifferenceCTE as (

SELECT

w_job_f.job_id,

w_time_d.time_id,

w_job_f.date_promised,

w_location_d.location_id,

w_location_d.location_name,

w_sales_class_d.sales_class_id,

w_sales_class_d.sales_class_desc,

w_job_f.date_ship_by,

ShipmentDelaysCTE.Shipping_First_Date,

(getBusDaysDiff(ShipmentDelaysCTE.Shipping_First_Date,
w_job_f.DATE_SHIP_BY)) AS Business_Days_Difference

FROM ShipmentDelaysCTE

INNER JOIN w_job_f on w_job_f.job_id = ShipmentDelaysCTE.job_id

INNER JOIN w_time_d on w_time_d.time_id = w_job_f.date_promised

INNER JOIN w_location_d ON w_job_f.location_id = w_location_d.location_id

INNER JOIN w_sales_class_d ON w_job_f.sales_class_id =
w_sales_class_d.sales_class_id

WHERE

```

w_location_d.location_id = w_job_f.location_id
AND w_sales_class_d.sales_class_id = w_job_f.sales_class_id
AND ShipmentDelaysCTE.job_id = w_job_f.job_id
AND ShipmentDelaysCTE.Shipping_First_Date > w_job_f.DATE_SHIP_BY
GROUP BY
w_job_f.job_id,
w_time_d.time_id,
w_location_d.location_id,
w_location_d.location_name,
w_sales_class_d.sales_class_id,
w_sales_class_d.sales_class_desc,
w_job_f.date_ship_by,
ShipmentDelaysCTE.Shipping_First_Date
)

```

```

select DaysDifferenceCTE.location_name, w_time_d.time_year, sum(Business_Days_Difference)
as Sum_bus_Days_Diff,
rank() over (partition by w_time_d.time_year order by sum(Business_Days_Difference) desc),
dense_rank() over (partition by w_time_d.time_year order by sum(Business_Days_Difference)
desc),
percent_rank() over (partition by w_time_d.time_year order by sum(Business_Days_Difference)
desc)

```

```

from DaysDifferenceCTE, w_time_d
where DaysDifferenceCTE.time_id = w_time_d.time_id
group by DaysDifferenceCTE.location_name, w_time_d.time_year

```


pgAdmin 4

File Object Tools Help

Browser

Databases (8)

- jobdb
- patientdb
- public
- Aggregates
- Correlations
- Domains
- FTS Configurations
- FTS Dictionaries
- FTS Parsers
- FTS Templates
- Foreign Tables
- Functions
- Materialized Views
- Operators
- Procedures
- Sequences
- Tables (3)
- diagnodetrauma
 - Columns (6)
 - diagcode
 - diocese
 - diotype
 - diomechanism
 - diocoverweight
 - diocostraints
 - RLS Policies
 - Rules
 - Triggers
 - patient
 - Columns (4)
 - patid
 - patage
 - patsex
 - patstatus

Query

Query History

```
25 INNER JOIN w_sales_class_d ON w_job.f.sales_class_id = w_sales_class_d.sales_class_id
26 WHERE
27 w_location_d.location_id = w_job.f.location_id
28 AND w_sales_class_d.sales_class_id = w_job.f.sales_class_id
29 AND ShipmentDelaysCTE.job_id = w_job.f.job_id
30 AND ShipmentDelaysCTE.Shipping_First_Date > w_job.f.DATE_SHIP_BY
31 GROUP BY
32 w_job.f.job_id,
33 w_time_d.time_id,
34 w_location_d.location_id,
35 w_location_d.location_name,
36 w_sales_class_d.sales_class_id,
37 w_sales_class_d.sales_class_desc,
38 w_job.f.date_ship_by,
39 ShipmentDelaysCTE.Shipping_First_Date
40 )
41
42 select DaysDifferenceCTE.location_name, w_time_d.time_year, sum(Business_Days_Difference) as Sum_Bus_Days_Diff,
43 rank() over (partition by w_time_d.time_year order by sum(Business_Days_Difference) desc),
44 dense_rank() over (partition by w_time_d.time_year order by sum(Business_Days_Difference) desc),
45 percent_rank() over (partition by w_time_d.time_year order by sum(Business_Days_Difference) desc)
46
47 from DaysDifferenceCTE, w_time_d
48 where DaysDifferenceCTE.time_id = w_time_d.time_id
49 group by DaysDifferenceCTE.location_name, w_time_d.time_year
50
51
52
```

Data Output

Messages

Notifications

	location_name	time_year	sum_bus_days_diff	rank	dense_rank	percent_rank
	character varying (30)	integer	numeric	bigint	bigint	double precision
1	Seattle	2013	47	1	1	0
2	Los Angeles	2013	23	2	2	0.1
3	Toronto	2013	19	3	3	0.2
4	Montreal	2013	18	4	4	0.3
5	Dallas	2013	11	5	5	0.4
6	Chicago	2013	10	6	6	0.5
7	London	2013	9	7	7	0.6
8	Denver	2013	8	8	8	0.7
9	Birmingham	2013	4	9	9	0.8
10	Atlanta	2013	4	9	9	0.8

Total rows: 30 of 30 Query complete 00:00:00.079

Successfully run. Total query runtime: 79 msec. 30 rows affected.

AQ10:

WITH PairsCTE AS (

SELECT

w_job_f.job_id,
w_location_d.location_id,
w_location_d.location_name,
w_sales_class_d.sales_class_id,
w_sales_class_d.sales_class_desc,
w_job_f.date_promised,
max(actual_ship_date) as last_ship_date,
sum (actual_quantity) as after_ship_quantity_sum,
w_job_f.quantity_ordered,

(getBusDaysDiff(actual_ship_date,date_promised)) AS
Business_Days_Difference,
w_sub_job_f.job_id as sub_job_id

FROM

w_job_f,
w_sales_class_d,
w_location_d,
w_sub_job_f,
w_job_shipment_f

WHERE

w_job_f.job_id = w_sub_job_f.job_id
AND w_sales_class_d.sales_class_id = w_job_f.sales_class_id
AND w_location_d.location_id = w_job_f.location_id
AND w_sub_job_f.sub_job_id = w_job_shipment_f.sub_job_id
AND actual_ship_date > date_promised

GROUP BY

w_sub_job_f.sub_job_id,
 w_job_f.job_id,
 w_location_d.location_id,
 w_location_d.location_name,
 w_sales_class_d.sales_class_id,
 w_sales_class_d.sales_class_desc,
 Business_Days_Difference

)

SELECT location_name, w_time_d.time_year, count(after_ship_quantity_sum) as
 count_delayed_jobs, sum(Business_Days_Difference) as Sum_dif_business_days,

sum(Quantity_Ordered - after_ship_quantity_sum)/sum(last_ship_date) as
 on_time_rate,

rank() over (partition by time_year order by sum(Quantity_Ordered -
 after_ship_quantity_sum)/sum(last_ship_date) desc)

FROM PairsCTE, w_time_d

group by location_name, time_year;

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

```

17 w_job_f,
18 w_sales_class_d,
19 w_location_d,
20 w_sub_job_f,
21 w_job_shipment_f,
22
23 WHERE
24   w_job_f.job_id = w_sub_job_f.job_id
25   AND w_sales_class_d.sales_class_id = w_job_f.sales_class_id
26   AND w_location_d.location_id = w_job_f.location_id
27   AND w_sub_job_f.sub_job_id = w_job_shipment_f.sub_job_id
28   AND actual_ship_date > date_promised
29
30 GROUP BY
31   w_sub_job_f.sub_job_id,
32   w_job_f.job_id,
33   w_location_d.location_id,
34   w_location_d.location_name,
35   w_sales_class_d.sales_class_id,
36   w_sales_class_d.sales_class_desc,
37   Business_Days_Difference
38
39 SELECT location_name, w_time_d.time_year, count(after_ship_quantity_sum) as count_delayed_jobs, sum(Business_Days_Difference) as Sum_dif_business_days,
40 sum(Quantity_Ordered - after_ship_quantity_sum)/sum(last_ship_date) as on_time_rate,
41 rank() over (partition by time_year order by sum(Quantity_Ordered - after_ship_quantity_sum)/sum(last_ship_date) desc )
42 FROM PairsCTE, w_time_d
43 group by location_name, time_year;
44

```

The query results are displayed in a table with the following columns: location_name, time_year, count_delayed_jobs, sum_dif_business_days, on_time_rate, and rank. The results show data for various locations including Dallas, Montreal, Birmingham, Seattle, Vancouver, London, Chicago, Toronto, Los Angeles, and Denver.

location_name	time_year	count_delayed_jobs	sum_dif_business_days	on_time_rate	rank
Dallas	2012	4176	7569	0.03088933208978162	1
Montreal	2012	12006	46414	0.04812431728886389076	2
Birmingham	2012	6786	26361	0.04720594920289592426	3
Seattle	2012	19314	64206	0.0419135676290784231	4
Vancouver	2012	9918	34713	0.03749620458242623300	5
London	2012	10962	22185	0.03488528454685491170	6
Chicago	2012	7569	14356	0.0348107264265029291	7
Toronto	2012	5203	15660	0.03041392622092378912	8
Los Angeles	2012	12769	48807	0.02916393797525291704	9
Denver	2012	2871	5491	0.0286510120799437122	10

Total rows: 72 of 72 Query complete 00:00:00.366