Appendices - Advanced exercises

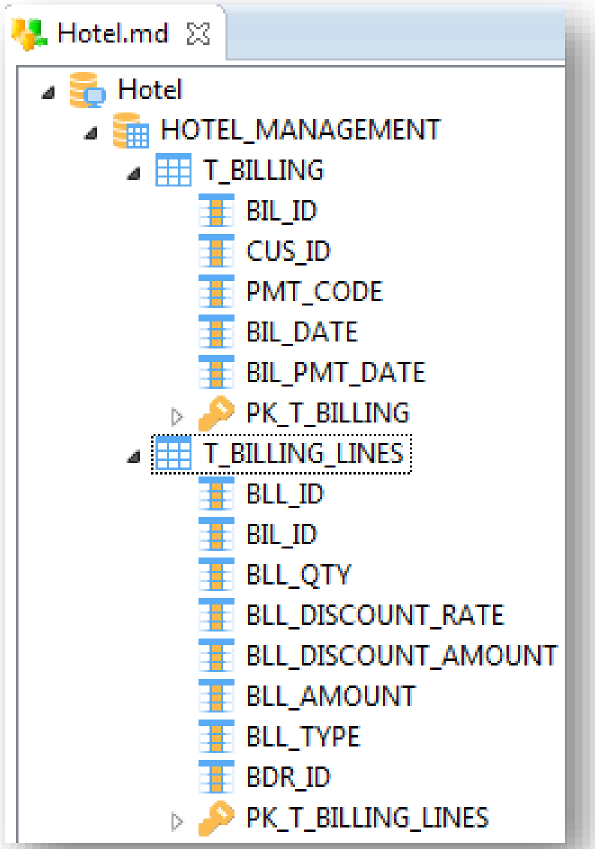
ADV01 Pivot

Rows in columns and inverse

Pivot - 1/6

Context

- Sometimes, you need to pivot data stored in rows to columns or inverse
- To be able to show this, we will used two sources tables of Hotel_management
  - T_BILLING
  - T_BILLING_LINES
    - The values are stored in rows in this table

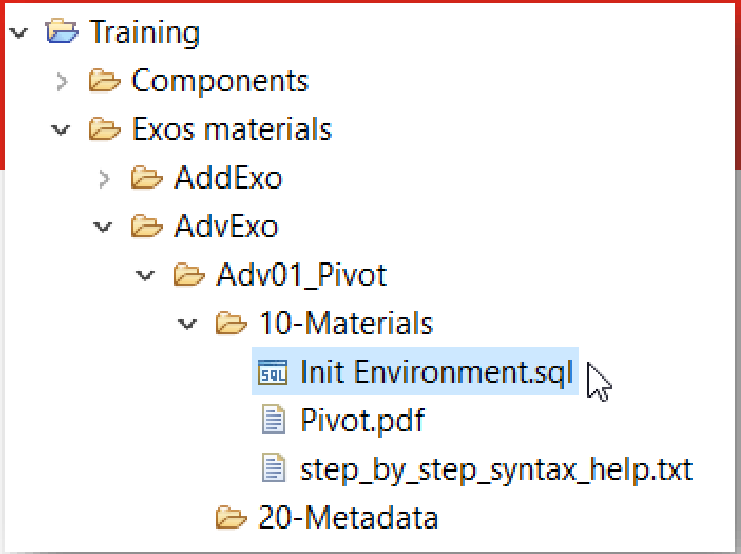


```
Hotel.md
Hotel
  HOTEL_MANAGEMENT
    T_BILLING
      BIL_ID
      CUS_ID
      PMT_CODE
      BIL_DATE
      BIL_PMT_DATE
      PK_T_BILLING
    T_BILLING_LINES
      BLL_ID
      BIL_ID
      BLL_QTY
      BLL_DISCOUNT_RATE
      BLL_DISCOUNT_AMOUNT
      BLL_AMOUNT
      BLL_TYPE
      BDR_ID
      PK_T_BILLING_LINES
```

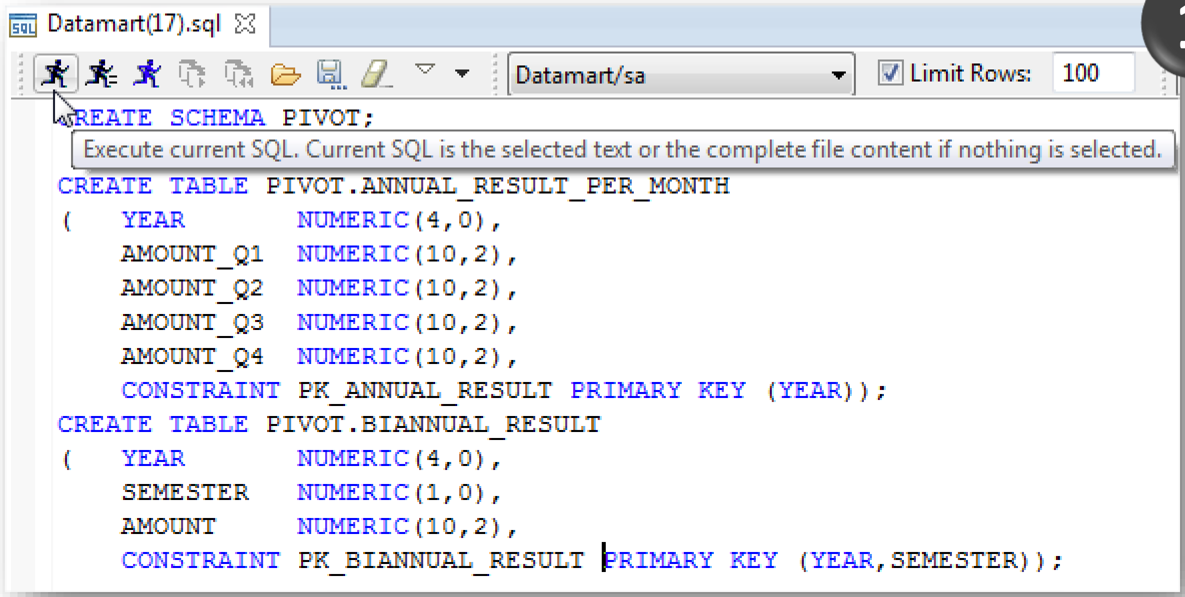
NEED HELP? Training/Exos materials/AdvExo/Adv01_.../step_by_step_syntax_help.txt

Pivot - 2/6

- 1 - Execute the SQL file (Init Environment.sql) on DATAMART connection to create the required schema and tables on a Datamart connection



1



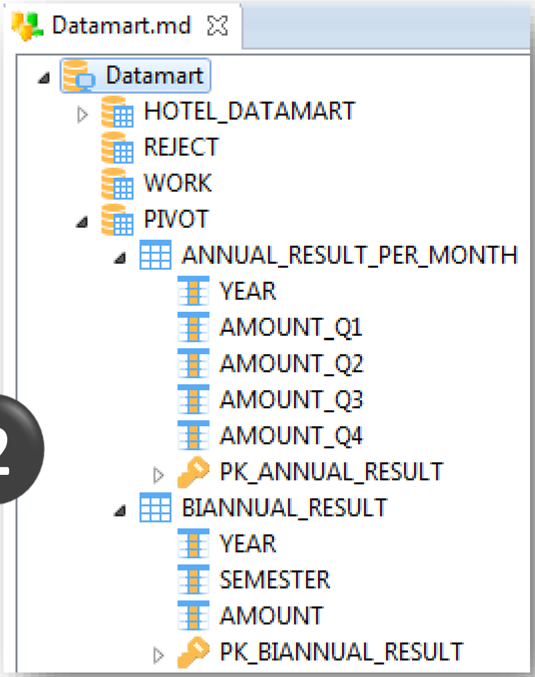
Execute current SQL. Current SQL is the selected text or the complete file content if nothing is selected.

```
CREATE SCHEMA PIVOT;

CREATE TABLE PIVOT.ANNUAL_RESULT_PER_MONTH
(   YEAR        NUMERIC(4,0),
    AMOUNT_Q1   NUMERIC(10,2),
    AMOUNT_Q2   NUMERIC(10,2),
    AMOUNT_Q3   NUMERIC(10,2),
    AMOUNT_Q4   NUMERIC(10,2),
    CONSTRAINT PK_ANNUAL_RESULT PRIMARY KEY (YEAR));
CREATE TABLE PIVOT.BIANNUAL_RESULT
(   YEAR        NUMERIC(4,0),
    SEMESTER    NUMERIC(1,0),
    AMOUNT      NUMERIC(10,2),
    CONSTRAINT PK_BIANNUAL_RESULT PRIMARY KEY (YEAR,SEMESTER));
```

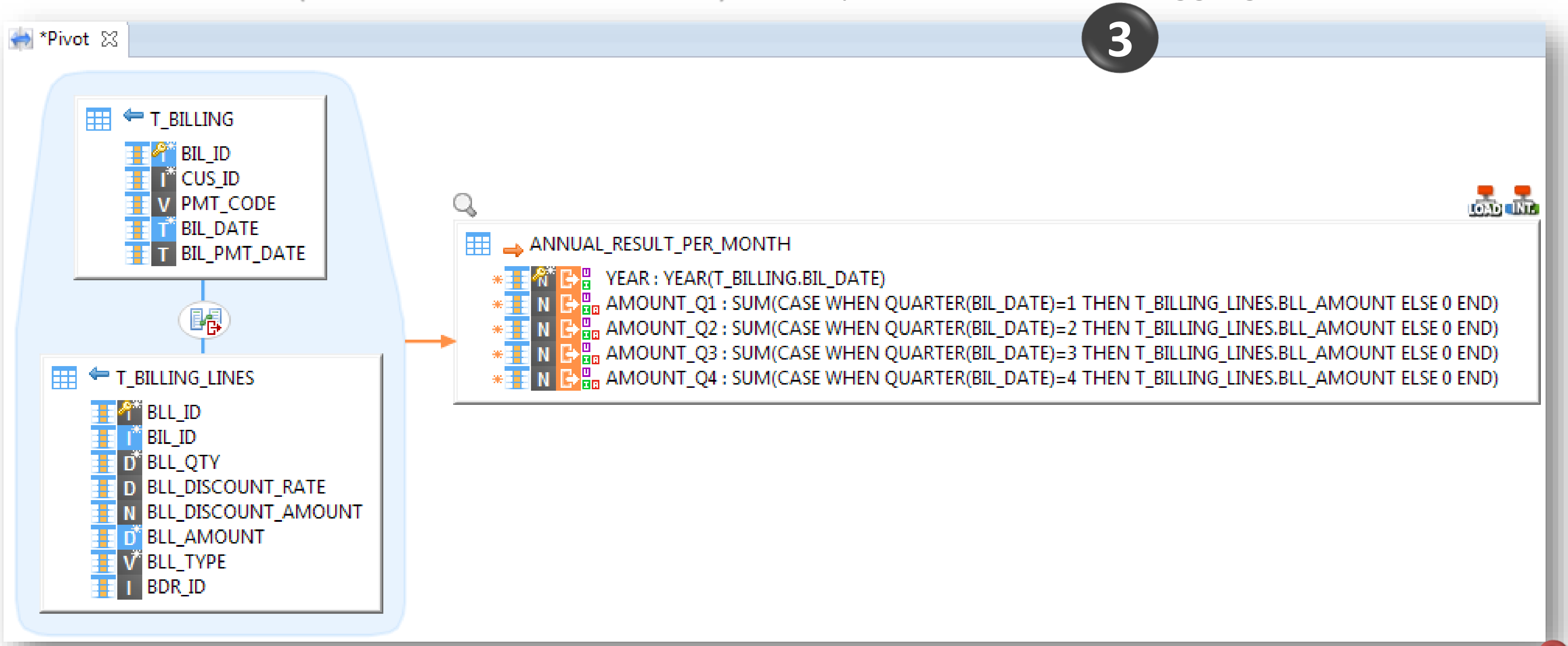
- 2 - Reverse the schema and the tables

2



Pivot - 3/6

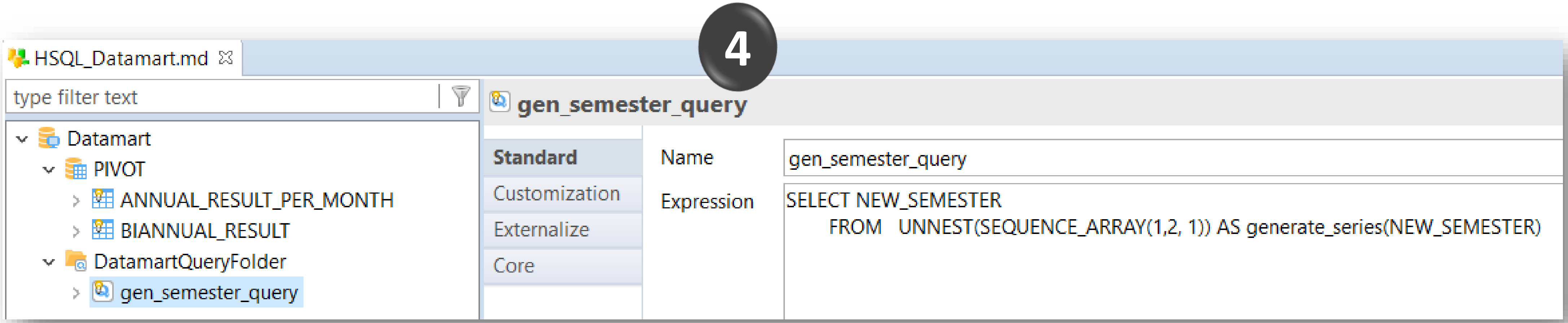
- 3 - Create a mapping. The first part will be to set rows data in columns (pivot rows to columns)
  - Depending on the quarter of each BIL_DATE, we will add the amount of the billing_lines in different columns
    - with **SUM(CASE WHEN … ELSE 0 END)** SQL expressions to set as "Aggregate"



YEAR : YEAR(T_BILLING.BIL_DATE)
AMOUNT_Q1 : SUM(CASE WHEN QUARTER(BIL_DATE)=1 THEN T_BILLING_LINES.BLL_AMOUNT ELSE 0 END)
AMOUNT_Q2 : SUM(CASE WHEN QUARTER(BIL_DATE)=2 THEN T_BILLING_LINES.BLL_AMOUNT ELSE 0 END)
AMOUNT_Q3 : SUM(CASE WHEN QUARTER(BIL_DATE)=3 THEN T_BILLING_LINES.BLL_AMOUNT ELSE 0 END)
AMOUNT_Q4 : SUM(CASE WHEN QUARTER(BIL_DATE)=4 THEN T_BILLING_LINES.BLL_AMOUNT ELSE 0 END)

Pivot - 4/6

- 4 - The second part of the mapping will be the use of quarter amounts (in columns) to generate rows for each semester
  - To be able to do this, we need a query that generate two semesters (“1” and “2” values)



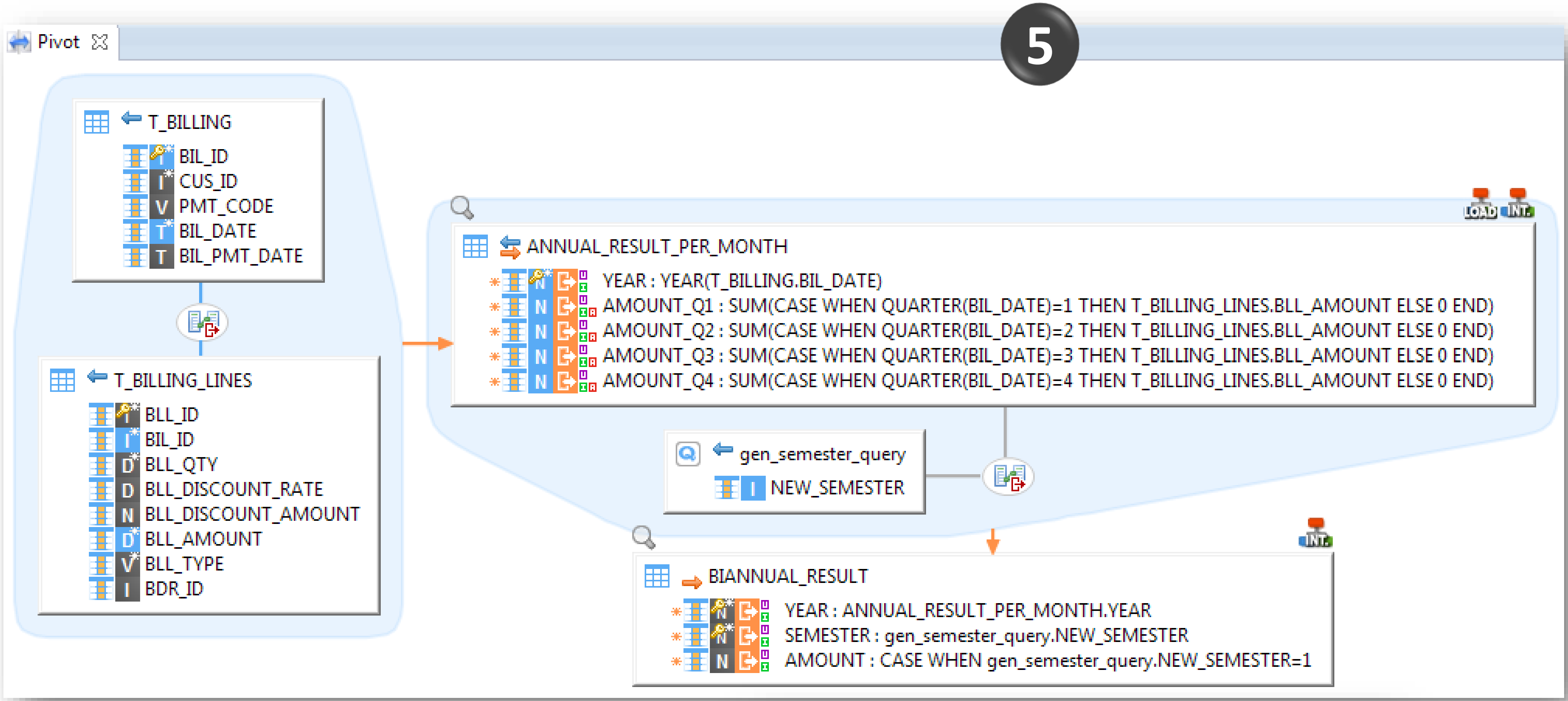
Name: gen_semester_query

Expression:

```
SELECT NEW_SEMESTER
    FROM   UNNEST(SEQUENCE_ARRAY(1,2, 1)) AS generate_series(NEW_SEMESTER)
```

Pivot - 5/6

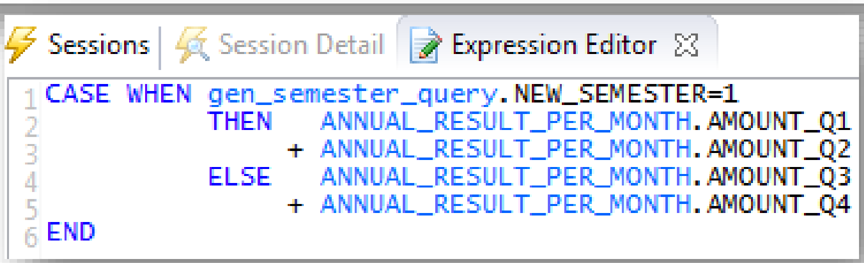
- 5 - Drag & Drop the query and choose a cross join with the ANNUAL_RESULT_PER_MONTH table



ANNUAL_RESULT_PER_MONTH
- YEAR : YEAR(T_BILLING.BIL_DATE)
- AMOUNT_Q1 : SUM(CASE WHEN QUARTER(BIL_DATE)=1 THEN T_BILLING_LINES.BLL_AMOUNT ELSE 0 END)
- AMOUNT_Q2 : SUM(CASE WHEN QUARTER(BIL_DATE)=2 THEN T_BILLING_LINES.BLL_AMOUNT ELSE 0 END)
- AMOUNT_Q3 : SUM(CASE WHEN QUARTER(BIL_DATE)=3 THEN T_BILLING_LINES.BLL_AMOUNT ELSE 0 END)
- AMOUNT_Q4 : SUM(CASE WHEN QUARTER(BIL_DATE)=4 THEN T_BILLING_LINES.BLL_AMOUNT ELSE 0 END)

BIANNUAL_RESULT
- YEAR : ANNUAL_RESULT_PER_MONTH.YEAR
- SEMESTER : gen_semester_query.NEW_SEMESTER
- AMOUNT : CASE WHEN gen_semester_query.NEW_SEMESTER=1

- The SQL Expression for AMOUNT



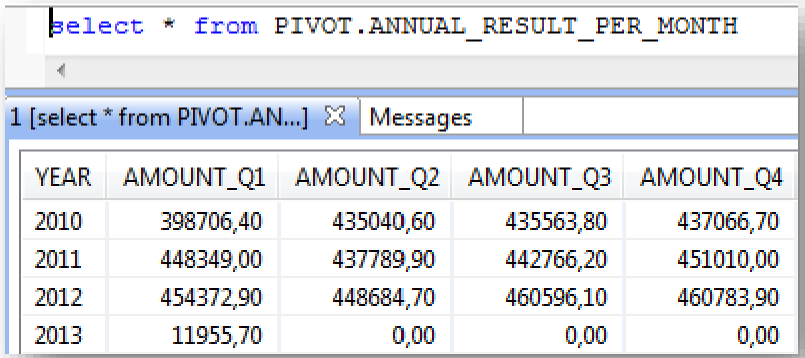
```
CASE WHEN gen_semester_query.NEW_SEMESTER=1
          THEN    ANNUAL_RESULT_PER_MONTH.AMOUNT_Q1
                + ANNUAL_RESULT_PER_MONTH.AMOUNT_Q2
          ELSE    ANNUAL_RESULT_PER_MONTH.AMOUNT_Q3
                + ANNUAL_RESULT_PER_MONTH.AMOUNT_Q4
END
```

Pivot - 6/6

6 - Analysis of the results and pivot display

Pivot rows → columns

Pivot columns → rows

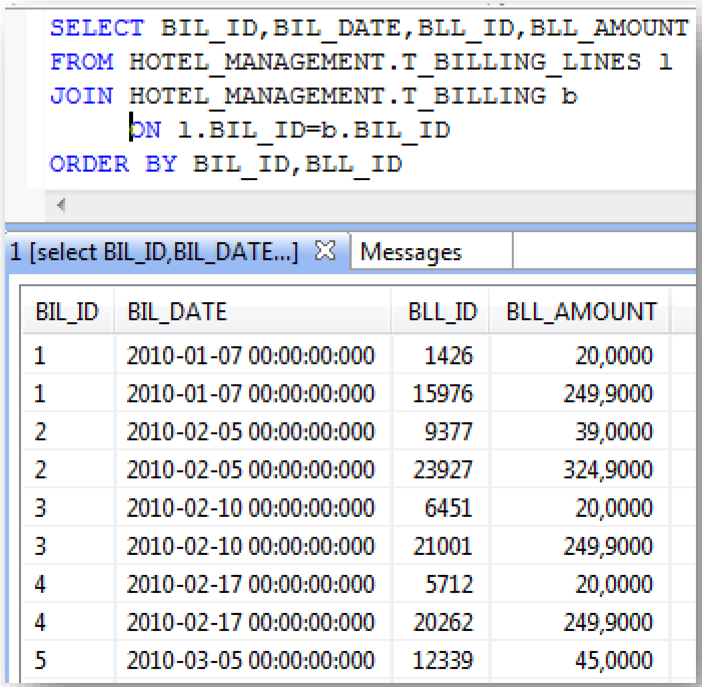


```
select * from PIVOT.ANNUAL_RESULT_PER_MONTH
```

1 [select * from PIVOT.AN...] | Messages

YEAR	AMOUNT_Q1	AMOUNT_Q2	AMOUNT_Q3	AMOUNT_Q4
2010	398706,40	435040,60	435563,80	437066,70
2011	448349,00	437789,90	442766,20	451010,00
2012	454372,90	448684,70	460596,10	460783,90
2013	11955,70	0,00	0,00	0,00

Amounts aggregated in columns

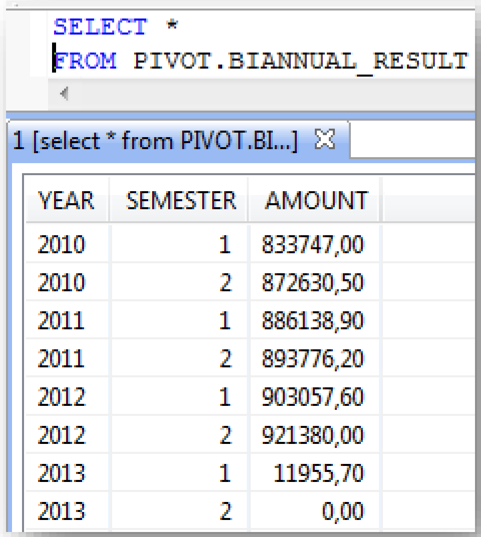


```
SELECT BIL_ID,BIL_DATE,BLL_ID,BLL_AMOUNT
FROM HOTEL_MANAGEMENT.T_BILLING_LINES l
JOIN HOTEL_MANAGEMENT.T_BILLING b
     ON l.BIL_ID=b.BIL_ID
ORDER BY BIL_ID,BLL_ID
```

1 [select BIL_ID,BIL_DATE...] | Messages

BIL_ID	BIL_DATE	BLL_ID	BLL_AMOUNT
1	2010-01-07 00:00:00:000	1426	20,0000
1	2010-01-07 00:00:00:000	15976	249,9000
2	2010-02-05 00:00:00:000	9377	39,0000
2	2010-02-05 00:00:00:000	23927	324,9000
3	2010-02-10 00:00:00:000	6451	20,0000
3	2010-02-10 00:00:00:000	21001	249,9000
4	2010-02-17 00:00:00:000	5712	20,0000
4	2010-02-17 00:00:00:000	20262	249,9000
5	2010-03-05 00:00:00:000	12339	45,0000

Amounts detailed in rows



```
SELECT *
FROM PIVOT.BIANNUAL_RESULT
```

1 [select * from PIVOT.BI...]

YEAR	SEMESTER	AMOUNT
2010	1	833747,00
2010	2	872630,50
2011	1	886138,90
2011	2	893776,20
2012	1	903057,60
2012	2	921380,00
2013	1	11955,70
2013	2	0,00

Amounts aggregated in rows