esProc

Some examples of solving BIRT dynamic data source

Issued by Raqsoft

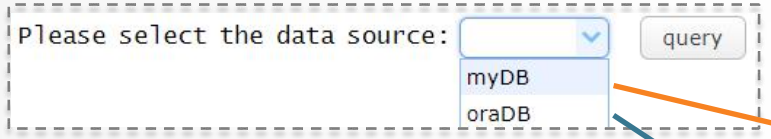
Contents

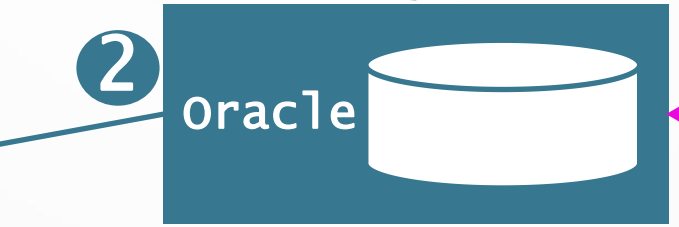
1. Parameter determines the connection of data source
2. Temporary splicing SQL and expression
3. Result set with indefinite number of columns
4. Group according to parameters

Parameter determines the connection of data source

Sometimes we need to specify the data source dynamically with parameters, or dynamically pass in the data source name to the sub-report or table control. Report tools such as BIRT must be implemented with high-level language, which is very troublesome.

The data sources myDB and oraDB point to different databases respectively. The two databases have the order table of same structure. The report needs to dynamically connect the data source according to the parameters and query the order with the amount greater than 1000 in the order table.





result

OrderID	Client	SellerId	Amount	OrderDate
2	UFS	13	1863	2009-07-...
5	DSG	15	3730	2009-07-...
6	JFE	10	1445	2009-07-...
8	PAER	29	2491	2010-07-...
10	JDR	17	1120	2010-07-...

result

OrderID	Client	SellerId	Amount	OrderDate
104	ERN	5	5678	2010-11-...
106	BSF	27	10742	2009-11-...
109	OFS	17	1106	2009-11-...

Order

OrderID	Client	SellerId	Amount	OrderDate
1	WVF	5	440	2009-02-...
2	UFS	13	1863	2009-07-...
4	JFS	27	671	2009-07-...
5	DSG	15	3730	2009-07-...
6	JFE	10	1445	2009-07-...
7	OLF	16	625	2009-07-...
8	PAER	29	2491	2010-07-...
9	DY	20	518	2010-07-...
10	JDR	17	1120	2010-07-...

Order

OrderID	Client	SellerId	Amount	OrderDate
101	WAN	22	396	2010-11-...
102	SPL	15	142	2010-11-...
103	LA	23	713	2010-11-...
104	ERN	5	5678	2010-11-...
105	FUR	28	154	2009-11-...
106	BSF	27	10742	2009-11-...
107	RHD	4	569	2009-11-...
108	BDR	12	480	2010-11-...
109	OFS	17	1106	2009-11-...

Parameter determines the connection of data source — Example

esProc supports dynamic analytic expressions, which can well meet this requirement.

	A	B
1	=${pSource}.query("select * from Order where Amount>?",pAmount)	/Receive parameters to query

pSource and pAmount are report parameters, where pSource is the data source name, ${…} represents resolving a string to an expression.

When pSource="myDB", the calculation results of A1 are shown in the right figure:

OrderID	Client	SellerId	Amount	OrderDate
2	UFS	13	1863	2009-07-...
5	DSG	15	3730	2009-07-...
6	IFE	10	1445	2009-07-...
8	PAER	29	2491	2010-07-...
10	IDR	17	1120	2010-07-...

When pSource="oraDB", the calculation results of A1 are shown in the right figure:

OrderID	Client	SellerId	Amount	OrderDate
104	ERN	5	5678	2010-11-...
106	BSF	27	10742	2009-11-...
109	OFS	17	1106	2009-11-...

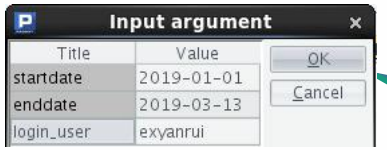
Temporary splicing SQL and expression — Permission query

According to the input user name, different order information is queried by level.

Author stores user department information, account stores account information (multiple users can belong to one account), meta stores order details.

result

Index	ITEMS	USERS	TOTALAMOUNT
31	sk316	tpbswww	1377783
32	sk316	tpbsxm	15226638
33	sk316	tpbsyy	215802
34	sk316	tpbszdw	207967
35	sk316	zqx12412	223605
36	vw421	epklcz	2230758
37	vw421	exCLY	1709520
38	vw421	exliujun	45077154
39	vw421	exlix	26648750
40	vw421	exwjs	14719705



Pass in the parameter login_user to determine the department level

author

Index	userid	dept
1	tpbrzwf	admin
2	exyanrui	manager
3	exzhlei	engineer

When dept=manager, get current account

account

Index	accountid	limits
1	vw421	exzhlei,tpbrzwf,exyanrui

When dept=admin， count sales of all accounts and users

Index	ITEMS	USERS	TOTALAMOUNT
1	vw421	epklcz	2230758
2	vw421	exCLY	1709520
3	vw421	exliujun	45077154
4	vw421	exlix	26648750
5	vw421	exwjs	14719705
6	vw421	exyanrui	27564393
7	vw421	exzhlei	17359915
8	vw421	gan60170	3069119
9	vw421	hd61861	4459959
10	vw421	hgz12422	1409795

When dept=manager, count sales of users under the current account

When dept=engineer, only count sales of user

Index	ITEMS	USERS	TOTALAMOUNT
1	vw421	exzhlei	17359915

META

Index	ORDERID	USERS	ITEMS	AMOUNT	STATE	ORDERDATE
1	94938.sm	tpbsxm	sk316	142037	0	2017-01-01 01:48:24
2	94936.sm	tpbsxm	sk316	168063	0	2017-01-01 08:55:49
3	95036.sm	tpbrextc	vw421	328236	0	2017-01-03 08:37:12
4	95039.sm	tpbrextc	vw421	332816	0	2017-01-03 09:53:42
5	95212.sm	tpbrextc	vw421	331207	0	2017-01-03 09:52:58
6	95093.sm	tpbrextc	vw421	332384	0	2017-01-03 09:53:47
7	95094.sm	tpbrextc	vw421	332243	0	2017-01-03 09:51:36
8	95213.sm	tpbrextc	vw421	331209	0	2017-01-03 09:53:10
9	95211.sm	tpbrextc	vw421	331194	0	2017-01-03 09:52:34
10	95092.sm	tpbrextc	vw421	332220	0	2017-01-03 09:50:53

Temporary splicing SQL and expression — Permission query — Example

The essence of the problem is to dynamically splice SQL conditions according to permission levels, which is very common in self-service query scenarios. The code using external program or stored procedure is hard to write and has a large amount of work; the logic judgment can be completed with less code by introducing esProc; its unique macro mechanism greatly improves the degree of code reuse.

	A	B	C
1	=connect("myDB")	=sql="\"select ITEMS,USERS,sum(AMOUNT) as TOTALAMOUNT from META where ORDERDATE>=? and ORDERDATE<=?"	/Define initial SQL
2	=flag=A1.query("select dept from author where userid=?",login_user).dept		/Query department with login user
3	if flag=="admin"	>B1=concat(sql," group by ITEMS,USERS\"")	/If admin, query all
4	else if flag=="manager"	>B1=concat(sql," and ITEMS='",A1.query("select accountid from account where find_in_set(?,limits)",login_user).accountid,"' group by ITEMS,USERS\"")	/If it is mananger, you can view the sales amount of each user under the current account
5	else	>B1=concat(sql," and USERS='",login_user,"' group by ITEMS,USERS\"")	/Ordinary users can only view their own sales
6	=A1.query@x(${B1},startdate,enddate)		/Execute spliced SQL

The spliced B1 SQL after B3 execution

Value
"select ITEMS,USERS,sum(AMOUNT) as TOTALAMOUNT from META where ORDERDATE>=? and ORDERDATE<=? group by ITEMS,USERS"

The spliced B1 SQL after B4 execution

Value
"select ITEMS,USERS,sum(AMOUNT) as TOTALAMOUNT from META where ORDERDATE>=? and ORDERDATE<=? and ITEMS='vw421' group by ITEMS,USERS"

The spliced B1 SQL after B5 execution

Value
"select ITEMS,USERS,sum(AMOUNT) as TOTALAMOUNT from META where ORDERDATE>=? and ORDERDATE<=? and USERS='exzhlei' group by ITEMS,USERS"

Temporary splicing SQL and expression— Splicing dynamic table name

Table A, B and C, where table A stores the table names of other tables, and queries the value of the corresponding ID in "tablename table" according to the tablename recorded in table A.

B table

Index	ID	Num
1	01	13
2	02	14
3	03	15

A table

Index	ID	TableName
1	01	b
2	02	c
3	03	b

C table

Index	ID	Num
1	01	91
2	02	92
3	03	93

result

Index	ID	val
1	01	13
2	02	92
3	03	15

It is very tedious to construct dynamic SQL with external programs. esProc can well support splicing operation.

	A	B
1	=connect("myDB")	/Connect database
2	=A1.query("select * from A")	/Query table A to form a sequence table
3	=A2.new(ID,A1.query("select * from "+TableName+" where ID ='"+ID+"'").Num:val)	/Dynamically spell SQL and query according to the tablename column of each record in A2

Temporary splicing SQL and expression — Query with in condition

Birt does not directly support array parameters to participate in queries, so it is difficult to process queries with in and cannot pass arrays to dataset to execute SQL with in.

In indirect mode, complex JS script needs to be written for SQL splicing, and it is very troublesome to decide whether to put quotation marks or not according to the data type of the in content.

eaProc can directly pass array parameters to SQL for query, and finally return the results to BIRT, which is very simple:

	A	B
1	=myDB.query("SELECT * FROM SALES WHERE ORDERID IN (?)",arg1.array())	/Pass arg1 array parameter, query data
2	return A1	/Return query results

Query results of A1

Index	ORDERID	CLIENT	SELLERID	AMOUNT	ORDERDATE
1	1	UJRNP	17	392.0	2012-11-02 ...
2	2	SJCH	6	4802.0	2012-11-09 ...
3	3	UJRNP	16	13500.0	2012-11-05 ...
4	4	PWQ	9	26100.0	2012-11-08 ...

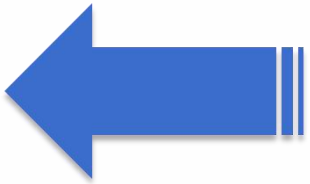
Argument to array

Temporary splicing SQL and expression — Dynamically analyze expression

In the TMP table of MySQL database, the value of size field is stored as a string, sometimes it is the product of three numbers, sometimes it is just a number. We want to convert this column to an integer value. MySQL can split strings very well, but it is difficult to calculate it.

result

Index	Size
1	150
2	20
3	3
4	60



Source data

Index	Size
1	15X10
2	5X4
3	3
4	2X6X5

For any four fundamental operations of arithmetic, operations with multiple parentheses and multiple operands, it is difficult to achieve with SQL. It can be solved by dynamic calculation in SPL:

	A	B
1	=myDB.query(“select * from tmp”)	/Query TMP table
2	>A1.run(Size=eval(replace(Size,"X","*")))	/Replace the character x with the multiplication symbol *, and eval function can dynamically parse the string into an expression and calculate

Result set with indefinite number of columns

According to the incoming time parameter, the time data from the beginning of the month to the current time is generated. For example, when the time parameter 1996-01-03 is imported, three columns (1996-01-01, 1996-01-02, 1996-01-03) are generated. When the time parameter 1996-01-08 is imported, eight date columns are generated.

result1

Index	O_ORDERSTATUS	O_ORDERPRIOR...	1992-01-01	1992-01-02	1992-01-03
1	F	1-URGENT	1255.35	4169.57	4193.39
2	F	2-HIGH	3778.66	9744.64	6329.68
3	F	3-MEDIUM	6484.37	1531.02	3124.04
4	F	4-NOT SPECIFIED	5038.66	5914.95	(null)
5	F	5-LOW	1253.62	3980.69	1463.16

Imported date parameter: 1996-01-03

Imported date parameter: 1996-01-08

result2

Index	O_ORDE...	O_ORDERPRIORI...	1992-01-01	1992-01-02	1992-01-03	1992-01-04	1992-01-05	1992-01-06	1992-01-07	1992-01-08
1	F	1-URGENT	1255.35	4169.57	4193.39	5538.6	7838.3	3008.37	4082.05	1816.65
2	F	2-HIGH	3778.66	9744.64	6329.68	5260.68	1587.26	8279.45	4737.0	3239.8
3	F	3-MEDIUM	6484.37	1531.02	3124.04	3724.23	8125.21	4037.81	1894.09	(null)
4	F	4-NOT SPECIFIED	5038.66	5914.95	1944.85	2429.96	2571.5	4746.42	4642.6	(null)
5	F	5-LOW	1253.62	3980.69	1463.16	3093.44	2323.7	3955.63	5837.97	2588.15

Source data

Index	O_ORDERKEY	O_ORDERSTATUS	O_ORDERPRIO...	O_ORDERDATE	O_TOTALPRICE
1	1	O	5-LOW	1996-01-02	1736.65
2	2	O	1-URGENT	1996-12-01	469.29
3	3	F	5-LOW	1993-10-14	1938.46
4	4	O	5-LOW	1995-10-11	321.52
5	5	F	5-LOW	1994-07-30	1446.59
6	6	F	4-NOT SPECIFI...	1992-02-21	587.5
7	7	O	2-HIGH	1996-01-10	2520.04
8	32	O	2-HIGH	1995-07-16	2086.61
9	33	F	3-MEDIUM	1993-10-27	1632.44
10	34	O	3-MEDIUM	1998-07-21	589.5
11	35	O	4-NOT SPECIFI...	1995-10-23	2537.25
12	36	O	1-URGENT	1995-11-03	682.9
13	37	F	3-MEDIUM	1992-06-03	2066.81
14	38	O	4-NOT SPECIFI...	1996-08-21	825.0
15	39	O	3-MEDIUM	1996-09-20	3417.34
16	64	F	3-MEDIUM	1994-07-16	394.15
17	65	P	1-URGENT	1995-03-18	1106.44
18	66	F	5-LOW	1994-01-20	1037.41
19	67	O	4-NOT SPECIFI...	1996-12-19	1694.05
20	68	O	3-MEDIUM	1998-04-18	3307.94

Result set with indefinite number of columns — Example

It is very difficult to write the result set of dynamic columns with SQL, and esProc can realize the dynamic effect flexibly, only need to write simple code.

	A	B
1	=myDB.query("select * from ORDERS where O_ORDERDATE>=? and O_ORDERDATE<=?",pdate@m(d_date),d_date)	/Filter query data set according to d_date parameter
2	=${"create(O_ORDERSTATUS,O_ORDERPRIORITY,"+periods(pdate@m(d_date),d_date,1).concat@c()+")"}	/Create an empty sequence table based on parameters
3	>A1.group(O_ORDERSTATUS,O_ORDERPRIORITY).run(A2.record(~.O_ORDERSTATUS | ~.O_ORDERPRIORITY | ~.group(O_ORDERDATE).(~.sum(O_TOTALPRICE))))	/Group and aggregate A1, and write the processed sequence result set to A2

Execution process of A3 grouping

Index	Member
1	[[1248,F,1-URGENT, ...],[3271,F,1-URGENT, ...],[36582,F,1-URGEN...
2	[[3712,F,2-HIGH, ...],[22693,F,2-HIGH, ...],[23010,F,2-HIGH, ...], ...]
3	[[3139,F,3-MEDIUM, ...],[29030,F,3-MEDIUM, ...],[37543,F,3-MEDIU...
4	[[5607,F,4-NOT SPECIFIED, ...],[6657,F,4-NOT SPECIFIED, ...],[20742...
5	[[24167,F,5-LOW, ...],[27015,F,5-LOW, ...],[51648,F,5-LOW, ...], ...]

Index	O_ORDERKEY	O_ORDERS...	O_ORDERP...	O_ORDERD...	O_TOTALPR...
1	1248	F	1-URGENT	1992-01-02	2620.81
2	3271	F	1-URGENT	1992-01-01	1255.35
3	36582	F	1-URGENT	1992-01-03	1869.33
4	47591	F	1-URGENT	1992-01-03	1360.57
5	69378	F	1-URGENT	1992-01-02	1548.76
6	83328	F	1-URGENT	1992-01-03	963.49

Index	O_ORDERKEY	O_ORDERS...	O_ORDERP...	O_ORDERD...	O_TOTALPR...
1	3712	F	2-HIGH	1992-01-02	1811.75
2	22693	F	2-HIGH	1992-01-03	2473.41
3	23010	F	2-HIGH	1992-01-01	973.73
4	27137	F	2-HIGH	1992-01-01	2804.93
5	63906	F	2-HIGH	1992-01-02	3285.97
6	80069	F	2-HIGH	1992-01-03	1405.16
7	90690	F	2-HIGH	1992-01-02	2605.68
8	108992	F	2-HIGH	1992-01-02	2041.24
9	112645	F	2-HIGH	1992-01-03	2451.11

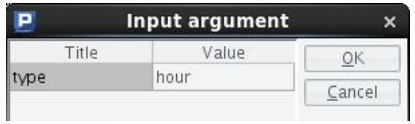
Group according to parameters — Statistical graph related operations

The X axis of the line graph is time scale, and different values should be displayed according to different parameter conditions. However, it is impossible to achieve the effect of changing the total data when the conditions are different on the birt statistical chart. Generally, external programs are needed to modify the scale of the x-axis, and the code is very cumbersome.

Parameter: day, display 24 hours, 1 hour interval

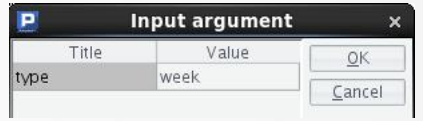
Index	Member	Value
1	2015-03-12 09:23:21	83.0625
2	2015-03-12 08:23:21	19.7053
3	2015-03-12 07:23:21	19.9969
4	2015-03-12 06:23:21	149.2472
5	2015-03-12 05:23:21	64.6114
6	2015-03-12 04:23:21	41.2842
7	2015-03-12 03:23:21	188.6375
8	2015-03-12 02:23:21	201.9394
9	2015-03-12 01:23:21	21.4358
10	2015-03-12 00:23:21	16.9647
11	2015-03-11 23:23:21	76.0822
12	2015-03-11 22:23:21	115.6825
13	2015-03-11 21:23:21	11.1431
14	2015-03-11 20:23:21	45.7211
15	2015-03-11 19:23:21	198.4781
16	2015-03-11 18:23:21	87.4881
17	2015-03-11 17:23:21	17.7075
18	2015-03-11 16:23:21	48.8608
19	2015-03-11 15:23:21	64.4744
20	2015-03-11 14:23:21	8.9958
21	2015-03-11 13:23:21	30.7856
22	2015-03-11 12:23:21	71.4450
23	2015-03-11 11:23:21	164.5139

Parameter: hour, display 12 items, interval 5 minutes



Index	Member	Value
1	2015-03-12 10:24:00	(null)
2	2015-03-12 10:19:00	0.9528
3	2015-03-12 10:14:00	(null)
4	2015-03-12 10:09:00	2.3806
5	2015-03-12 10:04:00	3.9444
6	2015-03-12 09:59:00	61.8811
7	2015-03-12 09:54:00	0.9956
8	2015-03-12 09:49:00	4.9547
9	2015-03-12 09:44:00	(null)
10	2015-03-12 09:39:00	5.6578
11	2015-03-12 09:34:00	2.2956
12	2015-03-12 09:29:00	(null)

Parameters: week, display 7 items, one item every 1 day



Index	Member	Value
1	2015-03-11 10:31:02	1767.9681
2	2015-03-10 10:31:02	1274.7014
3	2015-03-09 10:31:02	1905.5183
4	2015-03-08 10:31:02	2184.4608
5	2015-03-07 10:31:02	1985.4283
6	2015-03-06 10:31:02	2053.1322
7	2015-03-05 10:31:02	1872.7825

Group according to parameters — Statistical graph related operations — Example

The reporting tool is good at handling consistent rules, and for this inconsistent dynamic operation, the better way is to write a program to prepare the data source before drawing. esProc has rich set operations, and can easily complete such tasks.

	A	B	C
1	=[]		/Time interval data set to be generated
2	if(type==“hour”)	>A1=12.(elapse@s(now(),-5*60*~))	/Count the last hour, one item every 5 minutes, and generate 12 time intervals
3	else if(type==“day”)	>A1=24.(elapse@s(now(),-60*60*~))	/Count the last day, one item per hour, and generate 24 time intervals
4	else if(type==“week”)	>A1=7.(elapse(now(),-~))	/Count the last week, one item every day, and generate 7 time intervals
5	=A1.(demo.query(“select sum(v) from tv where t>? and t<=?”,~,ifn(~[-1],now()))).new(~.v:Value)		/According to the time interval, respectively query the statistical sum

A1 result after B2 execution

Index	Member
1	2015-03-12 10:24:00
2	2015-03-12 10:19:00
3	2015-03-12 10:14:00
4	2015-03-12 10:09:00
5	2015-03-12 10:04:00
6	2015-03-12 09:59:00
7	2015-03-12 09:54:00
8	2015-03-12 09:49:00
9	2015-03-12 09:44:00
10	2015-03-12 09:39:00
11	2015-03-12 09:34:00
12	2015-03-12 09:29:00

A1 result after B4 execution

Index	Member
1	2015-03-11 10:31:02
2	2015-03-10 10:31:02
3	2015-03-09 10:31:02
4	2015-03-08 10:31:02
5	2015-03-07 10:31:02
6	2015-03-06 10:31:02
7	2015-03-05 10:31:02

A1 result after B3 execution

Index	Member
1	2015-03-12 09:23:21
2	2015-03-12 08:23:21
3	2015-03-12 07:23:21
4	2015-03-12 06:23:21
5	2015-03-12 05:23:21
6	2015-03-12 04:23:21
7	2015-03-12 03:23:21
8	2015-03-12 02:23:21
9	2015-03-12 01:23:21
10	2015-03-12 00:23:21
11	2015-03-11 23:23:21
12	2015-03-11 22:23:21
13	2015-03-11 21:23:21
14	2015-03-11 20:23:21
15	2015-03-11 19:23:21
16	2015-03-11 18:23:21
17	2015-03-11 17:23:21
18	2015-03-11 16:23:21
19	2015-03-11 15:23:21
20	2015-03-11 14:23:21
21	2015-03-11 13:23:21
22	2015-03-11 12:23:21
23	2015-03-11 11:23:21

Group according to parameters — Dynamic date grouping

An enterprise report calculates orders within a period of time according to the start date and end date, but it needs to realize dynamic grouping by day, week, month and year according to the length of the period.

var diff = （end date – start date） -- date interval

Index	BeginDate	TotalOrder	TotalOrderAmount
1	2012-07-05	341	5498974.0
2	2013-07-05	491	8392346.0
3	2014-07-05	1	20400.0

1 if（diff>365）Group by year

Index	BeginDate	TotalOrder	TotalOrderAmount
1	2012-07-05	23	284883.0
2	2012-08-05	26	270621.0
3	2012-09-05	2	39596.0

2 if（diff>30）Group by month

Index	BeginDate	TotalOrder	TotalOrderAmount
1	2012-07-05	6	44951.0
2	2012-07-12	5	86892.0
3	2012-07-19	5	69952.0
4	2012-07-26	1	7719.0

3 if（diff>15）Group by week

Index	BeginDate	TotalOrder	TotalOrderAmount
1	2012-07-05	1	1161.0
2	2012-07-06	2	17300.0
3	2012-07-07	1	10260.0
4	2012-07-08	1	11634.0
5	2012-07-09	1	4596.0
6	2012-07-10	1	44499.0

4 if（diff<15）Group by day

ORDERS

Index	ORDERID	ORDERDATE	ORDERAMOUNT
1	10249	2012-07-05	1161.0
2	10250	2012-07-08	13166.0
3	10251	2012-07-08	4134.0
4	10252	2012-07-09	10260.0
5	10253	2012-07-10	11634.0
6	10254	2012-07-11	4596.0
7	10255	2012-07-12	44499.0
8	10256	2012-07-15	2794.0
9	10257	2012-07-16	24573.0
10	10258	2012-07-17	14051.0
11	10259	2012-07-18	975.0
12	10260	2012-07-19	5509.0
13	10261	2012-07-19	610.0
14	10262	2012-07-22	14487.0
15	10263	2012-07-23	43818.0
16	10264	2012-08-20	1101.0
17	10265	2012-07-25	5528.0
18	10266	2012-07-26	7719.0
19	10267	2012-07-29	20858.0
20	10268	2012-07-30	19887.0

Group according to parameters — Dynamic date grouping — Example

In essence, this problem is still data preparation, but in the way of SQL or scripted data sources, the code is hard to write and the workload is heavy; after introducing esProc into BIRT, its agile syntax system can complete the task with little code.

	A	B	C
1	=[]		/A1 stores the sequence of generated intervals
2	=demo.query@x("select ORDERID,ORDERDATE,ORDERAMOUNT from ORDERS where ORDERDATE>=? and ORDERDATE<=?",startDate,endDate)		/Query the order data between the statistics start date and end date from the orders table. StartDate and enddate are date parameters
3	=interval(startDate,endDate)		/Days between start and end dates
4	if A3>365	>A1=startDate|A3.(elapse@y(startDate,~))	/If the days interval is greater than 365 days, group by year
5	else if A3>30	>A1=startDate|A3.(elapse@m(startDate,~))	/Group by month if days interval is greater than 30 & less than or equal to 365 days
6	else if A3>15	>A1=startDate|A3.(elapse(startDate,7*~))	/Group by week if days interval is greater than 15 & less than or equal to 30 days
7	else	>A1=startDate|A3.(elapse(startDate,~))	/If the interval of days is less than 15 days, group by day
8	=A2.group(A1.pseg(ORDERDATE);~.count(ORDERID):TotalOrder,round(~.sum(ORDERAMOUNT),2):TotalOrderAmount,A1(#):BeginDate)		/Group A2 by A1 interval, count the total number of orders and the total amount of orders, keep two decimal places.
9	=A8.new(BeginDate:BeginDate,#2:TotalOrder,#3:TotalOrderAmount)		/Take out the data columns needed in A8 and generate a new result sequence table
10	return A9		/Return result set to BIRT

Innovation makes progress!