PIM Training Program

SQL

Subqueries and Wildcards

Learning Objective

- At the end of the module, the learner should be able
  - to write subqueries to avoid cross joints.
  - to run a single query with multiple wildcard criteria's

Agenda

- SQL
  - Subqueries
  - Avoiding 1-to-many joins
  - WITH clause
- ETL
  - More Wildcards
- Lesson 6: Assignment

Subqueries

Subqueries

- Queries can be nested within one another.
- Wrap the sub-query in parentheses and give it an alias, just like the way for a table.
- The subquery runs first, and the results are treated like a table in the outer query.
- Only the columns in the SELECT clause of the subquery will be available to the outer query.

Subqueries

An example

Write a query

```
SELECT
fcs.WAREHOUSE_ID as FCID
FROM D_WAREHOUSES fcs
WHERE fcs.REGION_ID = 4
```

Subqueries

An example

Wrap it in parentheses

```
(SELECT
fcs.WAREHOUSE_ID as FCID
FROM D_WAREHOUSES fcs
WHERE fcs.REGION_ID = 4
)
```

Subqueries

An example

```
(SELECT
fcs.WAREHOUSE_ID as FCID
FROM D_WAREHOUSES fcs
WHERE fcs.REGION_ID = 4
)
```

The results of this sub-query will be treated like a table with a column called FCID

FCID
BOM1

Subqueries

An example

```
(SELECT
fcs.WAREHOUSE_ID as FCID
FROM D_WAREHOUSES fcs
WHERE fcs.REGION_ID = 4
)a
```

Add “a” table alias

Subqueries

An example

Insert the sub query in the FROM clause of the main query

```
SELECT

FROM    (SELECT
        fcs.WAREHOUSE_ID as FCID
        FROM D_WAREHOUSES fcs
        WHERE fcs.REGION_ID = 4
        ) a
```

Subqueries

An Example

```
SELECT

FROM    (SELECT
        fcs.WAREHOUSE_ID as FCID
        FROM D_WAREHOUSES fcs
        WHERE fcs.REGION_ID = 4
        ) a
```

For the purposes of the outer query, ‘a’ is just a table with 1 column: FCID

Subqueries

An Example

```
SELECT
a.FCID
FROM    (SELECT
        fcs.WAREHOUSE_ID as FCID
        FROM D_WAREHOUSES fcs
        WHERE fcs.REGION_ID = 4
        ) a
```

Reference the columns from the results of the subquery in the SELECT clause of the outer query, using the sub-query's alias(a.FCID)

Subqueries

This is the completed Subquery

```
SELECT
a.FCID
FROM
(SELECT
fcs.WAREHOUSE_ID as FCID
FROM D_WAREHOUSES fcs
WHERE fcs.REGION_ID = 4
) a
```

SUBQUERY

Another Example

A real time query example to understand Sub queries

```
SELECT
promo.PROMO_ID
, SUM(CASE WHEN promo.PROMO_ID IS NOT NULL THEN 1 ELSE 0 END) as REDEMPTIONS
, COUNT(DISTINCT ucoi.CUSTOMER_ID) as CUSTOMERS
, COUNT(DISTINCT ucoi.ORDER_ID) as TOTALORDERS
, SUM(QUANTITY*OUR_PRICE) as OPS
, SUM(promo.FIRST_PROMO_AMT) as PROMO_AMT
, AVG(ucoi.OUR_PRICE) as ASP
, SUM(ucoi.QUANTITY) as UNITS
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 THEN ucoi.CUSTOMER_ID END) as CUST_1ORDER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS > 1 THEN ucoi.ORDER_ID END) as CUST_PROMOLATER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 AND ucoi.ORDER_DAY > promo.FIRST_ORDER_DAY THEN ucoi.ORDER_ID END) as CUST_NOPROMOLATER
FROM D_UNIFIED_CUSTOMER_ORDER_ITEMS ucoi
JOIN
        (SELECT CUSTOMER_ID
        , PROMO_ID
        , COUNT(DISTINCT ORDER_ID) as PROMOORDERS
        , MIN(ORDER_DAY) as FIRST_ORDER_DAY
        , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||ORDER_ID),18,19) as FIRST_ORDER_ID
        , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||TO_CHAR(ROUND(DISCOUNT_AMOUNT),'9999')),18, 8) as FIRST_PROMO_AMT
        FROM D_PROMOTION_ORDER_TXNS
        WHERE REGION_ID = 1
        AND MARKETPLACE_ID = 157860
        AND ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
        AND PROMO_ID IN (2229291,2229251,2335381,2365611,2361501,2413271)
        GROUP BY CUSTOMER_ID
        , PROMO_ID
        , ORDER_DAY
        ) promo
    ON ucoi.CUSTOMER_ID = promo.CUSTOMER_ID
WHERE
ucoi.REGION_ID = 1
AND ucoi.MARKETPLACE_ID = 157860
AND ucoi.ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
AND ucoi.ORDER_DAY >= promo.FIRST_ORDER_DAY
GROUP BY promo.PROMO_ID
;
```

The subquery, aliased as 'promo' runs first – producing aggregated results by CUSTOMER_ID , PROMO_ID and ORDER_DAY

Another Example

```
SELECT
promo.PROMO_ID
, SUM(CASE WHEN promo.PROMO_ID IS NOT NULL THEN 1 ELSE 0 END) as REDEMPTIONS
, COUNT(DISTINCT ucoi.CUSTOMER_ID) as CUSTOMERS
, COUNT(DISTINCT ucoi.ORDER_ID) as TOTALORDERS
, SUM(QUANTITY*OUR_PRICE) as OPS
, SUM(promo.FIRST_PROMO_AMT) as PROMO_AMT
, AVG(ucoi.OUR_PRICE) as ASP
, SUM(ucoi.QUANTITY) as UNITS
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 THEN ucoi.CUSTOMER_ID END) as CUST_1ORDER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS > 1 THEN ucoi.ORDER_ID END) as CUST_PROMOLATER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 AND ucoi.ORDER_DAY > promo.FIRST_ORDER_DAY THEN ucoi.ORDER_ID END) as CUST_NOPROMOLATER
FROM D_UNIFIED_CUSTOMER_ORDER_ITEMS ucoi
JOIN
                    (SELECT CUSTOMER_ID
                    , PROMO_ID
                    , COUNT(DISTINCT ORDER_ID) as PROMOORDERS
                    , MIN(ORDER_DAY) as FIRST_ORDER_DAY
                    , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||ORDER_ID),18,19) as FIRST_ORDER_ID
                    , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||TO_CHAR(ROUND(DISCOUNT_AMOUNT),'9999')),18, 8) as FIRST_PROMO_AMT
                    FROM D_PROMOTION_ORDER_TXNS
                    WHERE REGION_ID = 1
                    AND MARKETPLACE_ID = 157860
                    AND ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
                    AND PROMO_ID IN (2229291,2229251,2335381,2365611,2361501,2413271)
                    GROUP BY CUSTOMER_ID
                    , PROMO_ID
                    , ORDER_DAY
                    ) promo
          ON ucoi.CUSTOMER_ID = promo.CUSTOMER_ID
WHERE
ucoi.REGION_ID = 1
AND ucoi.MARKETPLACE_ID = 157860
AND ucoi.ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND
AND ucoi.ORDER_DAY >= promo.FIRST_ORDER_DAY
GROUP BY promo.PROMO_ID
;
```

The results of the subquery are then stored in TEMP space, and joined to the table D_UNIFIED_CUSTOMER_ORDER_ITEMS on CUSTOMER_ID

Another Example

```
SELECT
promo.PROMO_ID
, SUM(CASE WHEN promo.PROMO_ID IS NOT NULL THEN 1 ELSE 0 END) as REDEMPTIONS
, COUNT(DISTINCT ucoi.CUSTOMER_ID) as CUSTOMERS
, COUNT(DISTINCT ucoi.ORDER_ID) as TOTALORDERS
, SUM(QUANTITY*OUR_PRICE) as OPS
, SUM(promo.FIRST_PROMO_AMT) as PROMO_AMT
, AVG(ucoi.OUR_PRICE) as ASP
, SUM(ucoi.QUANTITY) as UNITS
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 THEN ucoi.CUSTOMER_ID END) as CUST_1ORDER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS > 1 THEN ucoi.ORDER_ID END) as CUST_PROMOLATER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 AND ucoi.ORDER_DAY > promo.FIRST_ORDER_DAY THEN ucoi.ORDER_ID END) as
CUST_NOPROMOLATER
FROM D_UNIFIED_CUSTOMER_ORDER_ITEMS ucoi
JOIN promo
        ON ucoi.CUSTOMER_ID = promo.CUSTOMER_ID
WHERE
ucoi.REGION_ID = 1
AND ucoi.MARKETPLACE_ID = 157860
AND ucoi.ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
AND ucoi.ORDER_DAY >= promo.FIRST_ORDER_DAY
GROUP BY promo.PROMO_ID
;
```

Almost like the results of subquery ‘promo’ are a table in DW called promo.

Another Example

```
SELECT
promo.PROMO_ID
, SUM(CASE WHEN promo.PROMO_ID IS NOT NULL THEN 1 ELSE 0 END) as REDEMPTIONS
, COUNT(DISTINCT ucoi.CUSTOMER_ID) as CUSTOMERS
, COUNT(DISTINCT ucoi.ORDER_ID) as TOTALORDERS
, SUM(QUANTITY*OUR_PRICE) as OPS
, SUM(promo.FIRST_PROMO_AMT) as PROMO_AMT
, AVG(ucoi.OUR_PRICE) as ASP
, SUM(ucoi.QUANTITY) as UNITS
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 THEN ucoi.CUSTOMER_ID END) as CUST_1ORDER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS > 1 THEN ucoi.ORDER_ID END) as CUST_PROMOLATER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 AND ucoi.ORDER_DAY > promo.FIRST_ORDER_DAY THEN ucoi.ORDER_ID END) as CUST_NOPROMOLATER
FROM D_UNIFIED_CUSTOMER_ORDER_ITEMS ucoi
JOIN
                    (SELECT CUSTOMER_ID
                    , PROMO_ID
                    , COUNT(DISTINCT ORDER_ID) as PROMOORDERS
                    , MIN(ORDER_DAY) as FIRST_ORDER_DAY
                    , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||ORDER_ID),18,19) as FIRST_ORDER_ID
                    , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||TO_CHAR(ROUND(DISCOUNT_AMOUNT),'9999')),18, 8) as FIRST_PROMO_AMT
                    FROM D_PROMOTION_ORDER_TXNS
                    WHERE REGION_ID = 1
                    AND MARKETPLACE_ID = 157860
                    AND ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
                    AND PROMO_ID IN (2229291,2229251,2335381,2365611,2361501,2413271)
                    GROUP BY CUSTOMER_ID
                    , PROMO_ID
                    , ORDER_DAY
                    ) promo
          ON ucoi.CUSTOMER_ID = promo.CUSTOMER_ID
WHERE
ucoi.REGION_ID = 1
AND ucoi.MARKETPLACE_ID = 157860
AND ucoi.ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
AND ucoi.ORDER_DAY >= promo.FIRST_ORDER_DAY
GROUP BY promo.PROMO_ID
;
```

The subquery is needed to get the MIN(ORDER_DAY) for each CUSTOMER_ID/PROMO_ID combination – so that we can use the result (now called FIRST_ORDER_DAY) in the outer query.

Another Example

```
SELECT
promo.PROMO_ID
, SUM(CASE WHEN promo.PROMO_ID IS NOT NULL THEN 1 ELSE 0 END) as REDEMPTIONS
, COUNT(DISTINCT ucoi.CUSTOMER_ID) as CUSTOMERS
, COUNT(DISTINCT ucoi.ORDER_ID) as TOTALORDERS
, SUM(QUANTITY*OUR_PRICE) as OPS
, SUM(promo.FIRST_PROMO_AMT) as PROMO_AMT
, AVG(ucoi.OUR_PRICE) as ASP
, SUM(ucoi.QUANTITY) as UNITS
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 THEN ucoi.CUSTOMER_ID END) as CUST_1ORDER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS > 1 THEN ucoi.ORDER_ID END) as CUST_PROMOLATER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 AND ucoi.ORDER_DAY > promo.FIRST_ORDER_DAY THEN ucoi.ORDER_ID END) as CUST_NOPROMOLATER
FROM D_UNIFIED_CUSTOMER_ORDER_ITEMS ucoi
JOIN
                    (SELECT CUSTOMER_ID
                    , PROMO_ID
                    , COUNT(DISTINCT ORDER_ID) as PROMOORDERS
                    , MIN(ORDER_DAY) as FIRST_ORDER_DAY
                    , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||ORDER_ID),18,19) as FIRST_ORDER_ID
                    , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||TO_CHAR(ROUND(DISCOUNT_AMOUNT),'9999')),18, 8) as FIRST_PROMO_AMT
                    FROM D_PROMOTION_ORDER_TXNS
                    WHERE REGION_ID = 1
                    AND MARKETPLACE_ID = 157860
                    AND ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
                    AND PROMO_ID IN (2229291,2229251,2335381,2365611,2361501,2413271)
                    GROUP BY CUSTOMER_ID
                    , PROMO_ID
                    , ORDER_DAY
                    ) promo
          ON ucoi.CUSTOMER_ID = promo.CUSTOMER_ID
WHERE
ucoi.REGION_ID = 1
AND ucoi.MARKETPLACE_ID = 157860
AND ucoi.ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
AND ucoi.ORDER_DAY >= promo.FIRST_ORDER_DAY
GROUP BY promo.PROMO_ID
;
```

Notice that any time a column from the subquery is referenced in the outer query, it is referenced by what that column is called in the results of the subquery, and NOT what the column might've been in the table in the subquery.

In this example, we use FIRST_PROMO_AMT and PROMOORDERS in the outer query, as these are the column aliases used in the subquery, and thus are what the columns are named in the temporary table created by the subquery.

Another Example

```
SELECT
promo.PROMO_ID
, SUM(CASE WHEN promo.PROMO_ID IS NOT NULL THEN 1 ELSE 0 END) as REDEMPTIONS
, COUNT(DISTINCT ucoi.CUSTOMER_ID) as CUSTOMERS
, COUNT(DISTINCT ucoi.ORDER_ID) as TOTALORDERS
, SUM(QUANTITY*OUR_PRICE) as OPS
, SUM(promo.FIRST_PROMO_AMT) as PROMO_AMT
, AVG(ucoi.OUR_PRICE) as ASP
, SUM(ucoi.QUANTITY) as UNITS
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 THEN ucoi.CUSTOMER_ID END) as CUST_1ORDER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS > 1 THEN ucoi.ORDER_ID END) as CUST_PROMOLATER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 AND ucoi.ORDER_DAY > promo.FIRST_ORDER_DAY THEN ucoi.ORDER_ID END) as CUST_NOPROMOLATER
FROM D_UNIFIED_CUSTOMER_ORDER_ITEMS ucoi
JOIN
                    (SELECT CUSTOMER_ID
                    , PROMO_ID
                    , COUNT(DISTINCT ORDER_ID) as PROMOORDERS
                    , MIN(ORDER_DAY) as FIRST_ORDER_DAY
                    , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||ORDER_ID),18,19) as FIRST_ORDER_ID
                    , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||TO_CHAR(ROUND(DISCOUNT_AMOUNT),'9999')),18, 8) as FIRST_PROMO_AMT
                    FROM D_PROMOTION_ORDER_TXNS
                    WHERE REGION_ID = 1
                    AND MARKETPLACE_ID = 157860
                    AND ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
                    AND PROMO_ID IN (2229291,2229251,2335381,2365611,2361501,2413271)
                    GROUP BY CUSTOMER_ID
                    , PROMO_ID
                    , ORDER_DAY
                    ) promo
          ON ucoi.CUSTOMER_ID = promo.CUSTOMER_ID
WHERE
ucoi.REGION_ID = 1
AND ucoi.MARKETPLACE_ID = 157860
AND ucoi.ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
AND ucoi.ORDER_DAY >= promo.FIRST_ORDER_DAY
GROUP BY promo.PROMO_ID
;
```

The outer query can only see columns in the table D_UNIFIED_CUSTOMER_ORDER_ITEMS and columns in the temporary table called ‘promo’.

Another Example

```
SELECT
promo.PROMO_ID
, SUM(CASE WHEN promo.PROMO_ID IS NOT NULL THEN 1 ELSE 0 END) as REDEMPTIONS
, COUNT(DISTINCT ucoi.CUSTOMER_ID) as CUSTOMERS
, COUNT(DISTINCT ucoi.ORDER_ID) as TOTALORDERS
, SUM(QUANTITY*OUR_PRICE) as OPS
, SUM(promo.FIRST_PROMO_AMT) as PROMO_AMT
, AVG(ucoi.OUR_PRICE) as ASP
, SUM(ucoi.QUANTITY) as UNITS
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 THEN ucoi.CUSTOMER_ID END) as CUST_1ORDER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS > 1 THEN ucoi.ORDER_ID END) as CUST_PROMOLATER
, COUNT(DISTINCT CASE WHEN promo.PROMOORDERS = 1 AND ucoi.ORDER_DAY > promo.FIRST_ORDER_DAY THEN ucoi.ORDER_ID END) as CUST_NOPROMOLATER
FROM D_UNIFIED_CUSTOMER_ORDER_ITEMS ucoi
JOIN
                    (SELECT CUSTOMER_ID
                    , PROMO_ID
                    , COUNT(DISTINCT ORDER_ID) as PROMOORDERS
                    , MIN(ORDER_DAY) as FIRST_ORDER_DAY
                    , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||ORDER_ID),18,19) as FIRST_ORDER_ID
                    , SUBSTR(MIN(TO_CHAR(ORDER_DAY,'YYYYMMDD HH24:MI:SS')||TO_CHAR(ROUND(DISCOUNT_AMOUNT),'9999')),18, 8) as FIRST_PROMO_AMT
                    FROM D_PROMOTION_ORDER_TXNS
                    WHERE REGION_ID = 1
                    AND MARKETPLACE_ID = 157860
                    AND ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
                    AND PROMO_ID IN (2229291,2229251,2335381,2365611,2361501,2413271)
                    GROUP BY CUSTOMER_ID
                    , PROMO_ID
                    , ORDER_DAY
                    ) promo
          ON ucoi.CUSTOMER_ID = promo.CUSTOMER_ID
WHERE
ucoi.REGION_ID = 1
AND ucoi.MARKETPLACE_ID = 157860
AND ucoi.ORDER_DAY BETWEEN TO_DATE('20110428','YYYYMMDD') AND TO_DATE('20110820','YYYYMMDD')
AND ucoi.ORDER_DAY >= promo.FIRST_ORDER_DAY
GROUP BY promo.PROMO_ID
;
```

For Example, We can't reference promo.MARKETPLACE_ID, because it's not in the SELECT clause of the promo subquery – and thus isn't in the temporary table called 'promo' that's created by the subquery… even though the column MARKETPLACE_ID is in the table D_PROMOTION_ORDER_TXNS that is used in the subquery.

A Complex Example

How many copies of Halloween Nation – 1589806808 (which includes an interview with your teacher) were ordered and shipped each day of the week beginning 9/18/2011 in the US?

A Complex Example

Let's start by looking at the Orders placed for this ASIN. The query below pulls the sum of daily ordered units for this ASIN from 18/09/2011 to 24/09/2011

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                     AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
;
```

A Complex example

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                     AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
;
```

Output

ACTIVITY_DAY	ORDER_UNITS
9/18/2011	3
9/19/2011	5
9/20/2011	7
9/21/2011	4
9/22/2011	3
9/23/2011	5
9/24/2011	2

A Complex example

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
;
```

ACTIVITY_DAY	ORDER_UNITS
9/18/2011	3
9/19/2011	5
9/20/2011	7
9/21/2011	4
9/22/2011	3
9/23/2011	5
9/24/2011	2

```
SELECT
dds.ACTIVITY_DAY
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
dds.ACTIVITY_DAY
;
```

And now lets look at the Shipped units per day from 18th to 24th Sept 2011 for the same ASIN

A Complex example

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
;
```

ACTIVITY_DAY	ORDER_UNITS
9/18/2011	3
9/19/2011	5
9/20/2011	7
9/21/2011	4
9/22/2011	3
9/23/2011	5
9/24/2011	2

```
SELECT
dds.ACTIVITY_DAY
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
dds.ACTIVITY_DAY
;
```

ACTIVITY_DAY	SHIP_UNITS
9/18/2011	7
9/19/2011	6
9/20/2011	7
9/21/2011	7
9/22/2011	3
9/23/2011	4
9/24/2011	2

Check

A Complex example

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
;
```

ACTIVITY_DAY	ORDER_UNITS
9/18/2011	3
9/19/2011	5
9/20/2011	7
9/21/2011	4
9/22/2011	3
9/23/2011	5
9/24/2011	2

```
SELECT
dds.ACTIVITY_DAY
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
dds.ACTIVITY_DAY
;
```

ACTIVITY_DAY	SHIP_UNITS
9/18/2011	7
9/19/2011	6
9/20/2011	7
9/21/2011	7
9/22/2011	3
9/23/2011	4
9/24/2011	2

ACTIVITY_DAY	ORDER_UNITS	SHIP_UNITS
9/18/2011	3	7
9/19/2011	5	6
9/20/2011	7	7
9/21/2011	4	7
9/22/2011	3	3
9/23/2011	5	4
9/24/2011	2	2

We can combine the results in Excel, but ultimately would like a single query to give us the combined results

A Complex example

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_ORDERS ddo
JOIN D_DAILY_SHIPMENTS dds
   ON ddo.ASIN = dds.ASIN
   AND ddo.ACTIVITY_DAY = dds.ACTIVITY_DAY
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                             AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
AND dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                             AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
;
```

We can start by using what we know about JOINs to combine the two queries

A Complex example

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_ORDERS ddo
JOIN D_DAILY_SHIPMENTS dds
   ON ddo.ASIN = dds.ASIN
   AND ddo.ACTIVITY_DAY = dds.ACTIVITY_DAY
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                          AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
AND dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                          AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
;
```

ACTIVITY_DAY	ORDER_UNITS	SHIP_UNITS
9/18/2011	21	7
9/19/2011	20	6
9/20/2011	42	14
9/21/2011	20	21
9/22/2011	9	6
9/23/2011	20	12
9/24/2011	4	6

But the results look all wrong

A Complex example

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_ORDERS ddo
JOIN D_DAILY_SHIPMENTS dds
   ON ddo.ASIN = dds.ASIN
   AND ddo.ACTIVITY_DAY = dds.ACTIVITY_DAY
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
AND dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
;
```

ACTIVITY_DAY	ORDER_UNITS	SHIP_UNITS
9/18/2011	3	7
9/19/2011	5	6
9/20/2011	7	7
9/21/2011	4	7
9/22/2011	3	3
9/23/2011	5	4
9/24/2011	2	2

ACTIVITY_DAY	ORDER_UNITS	SHIP_UNITS
9/18/2011	21	7
9/19/2011	20	6
9/20/2011	42	14
9/21/2011	20	21
9/22/2011	9	6
9/23/2011	20	12
9/24/2011	4	6

Comparing to the results of the two independently run queries that we combined in Excel shows we're overstating both ORDER_UNITS and SHIP_UNITS in many cases

A Complex example

Before concluding this example, we will look into the concept of 1 – to – many joins that will help you understand the solution better

Avoiding 1-to-many joins

1-to-many joins

When the ‘granularity’ of two data sets isn’t the same, it can lead to 1-to-many or even many-to-many joins.

These types of joins can result in SUM() and COUNT() functions over-stating their values.

Granularity

Defines at what level data is stored in the table.

Example 1: D_MP_ASINS_ESSENTIALS is at the ASIN/MARKETPLACE_ID grain, as the combination of those two defines a single unique record.

In other words, there are no two records in that table with equivalent values for both those columns.

Granularity

Defines at what level data is stored in the table.

Example 2: D_DISTRIBUTOR_ORDER_ITEMS is at the ORDER_ID/ISBN grain, as the combination of those two defines a single unique record.

There are no two records in that table with equivalent values for all 2 of those columns.

Granularity

What’s the Granularity of

D_WAREHOUSES?

D_MP_ASIN_BRAND_MANUFACTURER?

D_DISTRIBUTOR_ORDERS?

Granularity

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
;



SELECT
dds.ACTIVITY_DAY
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
dds.ACTIVITY_DAY
;
```

When joining tables, it's (often) necessary that they be at the same granularity. If they aren't, then you (may) have to force them to be before joining.

Granularity

```
SELECT
ddo.ACTIVITY_DAY
, ddo.ORDERED_UNITS as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
;



SELECT
dds.ACTIVITY_DAY
, dds.SHIPPED_UNITS as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
;
```

To understand the granularity of the tables we're working with, we'll remove the aggregation to view raw data

Granularity

```
SELECT
ddo.ACTIVITY_DAY
, ddo.ORDERED_UNITS as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
;



SELECT
dds.ACTIVITY_DAY
, dds.SHIPPED_UNITS as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
;
```

ACTIVITY_DAY	ORDER_UNITS
9/18/2011	3
9/19/2011	5
9/20/2011	6
9/20/2011	1
9/21/2011	2
9/21/2011	2
9/21/2011	0
9/22/2011	3
9/22/2011	0
9/23/2011	1
9/23/2011	4
9/23/2011	0
9/24/2011	1
9/24/2011	0
9/24/2011	1

Granularity

```
SELECT
ddo.ACTIVITY_DAY
, ddo.ORDERED_UNITS as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
;



SELECT
dds.ACTIVITY_DAY
, dds.SHIPPED_UNITS as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
;
```

ACTIVITY_DAY	ORDER_UNITS
9/18/2011	3
9/19/2011	5
9/20/2011	6
9/20/2011	1
9/21/2011	2
9/21/2011	2
9/21/2011	0
9/22/2011	3
9/22/2011	0
9/23/2011	1
9/23/2011	4
9/23/2011	0
9/24/2011	1
9/24/2011	0
9/24/2011	1

There are multiple records for most dates

Granularity

```
SELECT
ddo.ACTIVITY_DAY
, ddo.MERCHANT_CUSTOMER_ID
, ddo.ORDER_METHOD
, ddo.ORDERED_UNITS as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
;
```

```
SELECT
dds.ACTIVITY_DAY
, dds.SHIPPED_UNITS as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
;
```

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS
9/18/2011	608808520	S	3
9/19/2011	608808520	S	5
9/20/2011	608808520	S	6
9/20/2011	988123990	S	1
9/21/2011	608808520	1	2
9/21/2011	988123990	S	0
9/21/2011	608808520	S	2
9/22/2011	988123990	S	0
9/22/2011	608808520	S	3
9/23/2011	608808520	1	1
9/23/2011	988123990	S	0
9/23/2011	608808520	S	4
9/24/2011	608808520	1	1
9/24/2011	608808520	S	0
9/24/2011	608808520	O	1

There are multiple records for most dates, because the table stores one record per ASIN per MARKETPLACE_ID per ACTIVITY_DAY, per MERCHANT_CUSTOMER_ID per ORDER_METHOD, etc.

Granularity

```
SELECT
ddo.ACTIVITY_DAY
, ddo.MERCHANT_CUSTOMER_ID
, ddo.ORDER_METHOD
, ddo.ORDERED_UNITS as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
;



SELECT
dds.ACTIVITY_DAY
, dds.SHIPPED_UNITS as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
;
```

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS
9/18/2011	608808520	S	3
9/19/2011	608808520	S	5
9/20/2011	608808520	S	6
9/20/2011	988123990	S	1
9/21/2011	608808520	1	2
9/21/2011	988123990	S	0
9/21/2011	608808520	S	2
9/22/2011	988123990	S	0
9/22/2011	608808520	S	3
9/23/2011	608808520	1	1
9/23/2011	988123990	S	0
9/23/2011	608808520	S	4
9/24/2011	608808520	1	1
9/24/2011	608808520	S	0
9/24/2011	608808520	O	1

We can logically assume the same (or similar) is true for the D_DAILY_SHIPMENTS table

Granularity

```
SELECT
ddo.ACTIVITY_DAY
, ddo.MERCHANT_CUSTOMER_ID
, ddo.ORDER_METHOD
, ddo.ORDERED_UNITS as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
;
```

```
SELECT
dds.ACTIVITY_DAY
, dds.MERCHANT_CUSTOMER_ID
, dds.DISTRIBUTOR_ID
, dds.WAREHOUSE_ID
, dds.SHIPPED_UNITS as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
;
```

And it is

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	INJ	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	BTS	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	PELIC	BWI1	1
9/18/2011	608808520	PELIC	PHX3	1
9/18/2011	608808520	PELIC	PHX6	1
9/19/2011	608808520	PELIC	LEX1	3
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/20/2011	608808520	PELIC	LEX1	2
9/20/2011	608808520	PELIC	PHX6	1
9/20/2011	608808520	PELIC	PHL1	1
9/20/2011	608808520	BTBRD	BTBR	1
9/20/2011	608808520	INN	IND1	1
9/20/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHL1	2
9/21/2011	608808520	PELIC	PHX6	2
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	ING	SEA6	1
9/22/2011	988123990	-1	-1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	PHL1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	LEX1	1
9/24/2011	608808520	PELIC	PHX3	1
9/24/2011	608808520	PELIC	PHX6	1

Granularity

Every record for 9/18 from D_DAILY_ORDERS…

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS
9/18/2011	608808520	S	3
9/19/2011	608808520	S	5
9/20/2011	608808520	S	6
9/20/2011	988123990	S	1
9/21/2011	608808520	1	2
9/21/2011	988123990	S	0
9/21/2011	608808520	S	2
9/22/2011	988123990	S	0
9/22/2011	608808520	S	3
9/23/2011	608808520	1	1
9/23/2011	988123990	S	0
9/23/2011	608808520	S	4
9/24/2011	608808520	1	1
9/24/2011	608808520	S	0
9/24/2011	608808520	O	1

When we did a simple join between the two tables

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	INJ	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	BTS	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	PELIC	BWI1	1
9/18/2011	608808520	PELIC	PHX3	1
9/18/2011	608808520	PELIC	PHX6	1
9/19/2011	608808520	PELIC	LEX1	3
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/20/2011	608808520	PELIC	LEX1	2
9/20/2011	608808520	PELIC	PHX6	1
9/20/2011	608808520	PELIC	PHL1	1
9/20/2011	608808520	BTBRD	BTBR	1
9/20/2011	608808520	INN	IND1	1
9/20/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHL1	2
9/21/2011	608808520	PELIC	PHX6	2
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	ING	SEA6	1
9/22/2011	988123990	-1	-1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	PHL1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	LEX1	1
9/24/2011	608808520	PELIC	PHX3	1
9/24/2011	608808520	PELIC	PHX6	1

Granularity

Every record for 9/18 from D_DAILY_ORDERS…

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS
9/18/2011	608808520	S	3
9/19/2011	608808520	S	5
9/20/2011	608808520	S	6
9/20/2011	988123990	S	1
9/21/2011	608808520	1	2
9/21/2011	988123990	S	0
9/21/2011	608808520	S	2
9/22/2011	988123990	S	
9/22/2011	608808520	S	
9/23/2011	608808520	1	
9/23/2011	988123990	S	
9/23/2011	608808520	S	4
9/24/2011	608808520	1	1
9/24/2011	608808520	S	0
9/24/2011	608808520	O	1

Joined to every record for 9/18 from D_DAILY_SHIPMENTS

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	INJ	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	BTS	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	PELIC	BWI1	1
9/18/2011	608808520	PELIC	PHX3	1
9/18/2011	608808520	PELIC	PHX6	1
9/19/2011	608808520	PELIC	LEX1	3
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
	608808520	PELIC	LEX1	1
	608808520	PELIC	LEX1	2
	608808520	PELIC	PHX6	1
	608808520	PELIC	PHL1	1
	608808520	BTBRD	BTBR	1
9/20/2011	608808520	INN	IND1	1
9/20/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHL1	2
9/21/2011	608808520	PELIC	PHX6	2
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	ING	SEA6	1
9/22/2011	988123990	-1	-1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	PHL1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	LEX1	1
9/24/2011	608808520	PELIC	PHX3	1
9/24/2011	608808520	PELIC	PHX6	1

Creating a 1-to-Many join situation

When we did a simple join between the two tables

Granularity

And every record for 9/24 from D_DAILY_ORDERS…

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS
9/18/2011	608808520	S	3
9/19/2011	608808520	S	5
9/20/2011	608808520	S	6
9/20/2011	988123990	S	1
9/21/2011	608808520	1	2
9/21/2011	988123990	S	0
9/21/2011	608808520	S	2
9/22/2011	988123990	S	0
9/22/2011	608808520	S	3
9/23/2011	608808520	1	1
9/23/2011	988123990	S	0
9/23/2011	608808520	S	4
9/24/2011	608808520	1	1
9/24/2011	608808520	S	0
9/24/2011	608808520	O	1

When we did a simple join between the two tables

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	INJ	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	BTS	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	PELIC	BWI1	1
9/18/2011	608808520	PELIC	PHX3	1
9/18/2011	608808520	PELIC	PHX6	1
9/19/2011	608808520	PELIC	LEX1	3
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/20/2011	608808520	PELIC	LEX1	2
9/20/2011	608808520	PELIC	PHX6	1
9/20/2011	608808520	PELIC	PHL1	1
9/20/2011	608808520	BTBRD	BTBR	1
9/20/2011	608808520	INN	IND1	1
9/20/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHL1	2
9/21/2011	608808520	PELIC	PHX6	2
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	ING	SEA6	1
9/22/2011	988123990	-1	-1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	PHL1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	LEX1	1
9/24/2011	608808520	PELIC	PHX3	1
9/24/2011	608808520	PELIC	PHX6	1

Granularity

And every record for 9/24 from D_DAILY_ORDERS…

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS
9/18/2011	608808520	S	3
9/19/2011	608808520	S	5
9/20/2011	608808520	S	6
9/20/2011	988123990	S	1
9/21/2011	608808520	1	2
9/21/2011	988123990	S	0
9/21/2011	608808520	S	2
9/22/2011	988123990	S	
9/22/2011	608808520	S	
9/23/2011	608808520	1	
9/23/2011	988123990	S	
9/23/2011	608808520	S	4
9/24/2011	608808520	1	1
9/24/2011	608808520	S	0
9/24/2011	608808520	O	1

Joined to every record for 9/24 from D_DAILY_SHIPMENTS

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	INJ	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	BTS	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	PELIC	BWI1	1
9/18/2011	608808520	PELIC	PHX3	1
9/18/2011	608808520	PELIC	PHX6	1
9/19/2011	608808520	PELIC	LEX1	3
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
	608808520	PELIC	LEX1	1
	608808520	PELIC	LEX1	2
	608808520	PELIC	PHX6	1
	608808520	PELIC	PHL1	1
	608808520	BTBRD	BTBR	1
9/20/2011	608808520	INN	IND1	1
9/20/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHL1	2
9/21/2011	608808520	PELIC	PHX6	2
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	ING	SEA6	1
9/22/2011	988123990	-1	-1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	PHL1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	LEX1	1
9/24/2011	608808520	PELIC	PHX3	1
9/24/2011	608808520	PELIC	PHX6	1

Creating a Many-to-Many join situation

When we did a simple join between the two tables

Granularity

And so on…

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS
9/18/2011	608808520	S	3
9/19/2011	608808520	S	5
9/20/2011	608808520	S	6
9/20/2011	988123990	S	1
9/21/2011	608808520	1	2
9/21/2011	988123990	S	0
9/21/2011	608808520	S	2
9/22/2011	988123990	S	0
9/22/2011	608808520	S	3
9/23/2011	608808520	1	1
9/23/2011	988123990	S	0
9/23/2011	608808520	S	4
9/24/2011	608808520	1	1
9/24/2011	608808520	S	0
9/24/2011	608808520	O	1

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	INJ	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	BTS	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	PELIC	BWI1	1
9/18/2011	608808520	PELIC	PHX3	1
9/18/2011	608808520	PELIC	PHX6	1
9/19/2011	608808520	PELIC	LEX1	3
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/20/2011	608808520	PELIC	LEX1	2
9/20/2011	608808520	PELIC	PHX6	1
9/20/2011	608808520	PELIC	PHL1	1
9/20/2011	608808520	BTBRD	BTBR	1
9/20/2011	608808520	INN	IND1	1
9/20/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHL1	2
9/21/2011	608808520	PELIC	PHX6	2
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	ING	SEA6	1
9/22/2011	988123990	-1	-1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	PHL1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	LEX1	1
9/24/2011	608808520	PELIC	PHX3	1
9/24/2011	608808520	PELIC	PHX6	1

Granularity

And so on…

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS
9/18/2011	608808520	S	3
9/19/2011	608808520	S	5
9/20/2011	608808520	S	6
9/20/2011	988123990	S	1
9/21/2011	608808520	1	2
9/21/2011	988123990	S	0
9/21/2011	608808520	S	2
9/22/2011	988123990	S	0
9/22/2011	608808520	S	3
9/23/2011	608808520	1	1
9/23/2011	988123990	S	0
9/23/2011	608808520	S	4
9/24/2011	608808520	1	1
9/24/2011	608808520	S	0
9/24/2011	608808520	O	1

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	INJ	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	BTS	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	PELIC	BWI1	1
9/18/2011	608808520	PELIC	PHX3	1
9/18/2011	608808520	PELIC	PHX6	1
9/19/2011	608808520	PELIC	LEX1	3
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/20/2011	608808520	PELIC	LEX1	2
9/20/2011	608808520	PELIC	PHX6	1
9/20/2011	608808520	PELIC	PHL1	1
9/20/2011	608808520	BTBRD	BTBR	1
9/20/2011	608808520	INN	IND1	1
9/20/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHL1	2
9/21/2011	608808520	PELIC	PHX6	2
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	ING	SEA6	1
9/22/2011	988123990	-1	-1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	PHL1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	LEX1	1
9/24/2011	608808520	PELIC	PHX3	1
9/24/2011	608808520	PELIC	PHX6	1

Granularity

And so on.

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS
9/18/2011	608808520	S	3
9/19/2011	608808520	S	5
9/20/2011	608808520	S	6
9/20/2011	988123990	S	1
9/21/2011	608808520	1	2
9/21/2011	988123990	S	0
9/21/2011	608808520	S	2
9/22/2011	988123990	S	0
9/22/2011	608808520	S	3
9/23/2011	608808520	1	1
9/23/2011	988123990	S	0
9/23/2011	608808520	S	4
9/24/2011	608808520	1	1
9/24/2011	608808520	S	0
9/24/2011	608808520	O	1

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	INJ	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	BTS	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	PELIC	BWI1	1
9/18/2011	608808520	PELIC	PHX3	1
9/18/2011	608808520	PELIC	PHX6	1
9/19/2011	608808520	PELIC	LEX1	3
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/20/2011	608808520	PELIC	LEX1	2
9/20/2011	608808520	PELIC	PHX6	1
9/20/2011	608808520	PELIC	PHL1	1
9/20/2011	608808520	BTBRD	BTBR	1
9/20/2011	608808520	INN	IND1	1
9/20/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHL1	2
9/21/2011	608808520	PELIC	PHX6	2
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	ING	SEA6	1
9/22/2011	988123990	-1	-1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	PHL1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	LEX1	1
9/24/2011	608808520	PELIC	PHX3	1
9/24/2011	608808520	PELIC	PHX6	1

Granularity

SQL performs JOINs prior to the aggregation

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS
9/18/2011	608808520	S	3
9/19/2011	608808520	S	5
9/20/2011	608808520	S	6
9/20/2011	988123990	S	1
9/21/2011	608808520	1	2
9/21/2011	988123990	S	0
9/21/2011	608808520	S	2
9/22/2011	988123990	S	0
9/22/2011	608808520	S	3
9/23/2011	608808520	1	1
9/23/2011	988123990	S	0
9/23/2011	608808520	S	4
9/24/2011	608808520	1	1
9/24/2011	608808520	S	0
9/24/2011	608808520	O	1

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	INJ	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	BTS	ABE2	1
9/18/2011	608808520	INN	LEX1	1
9/18/2011	608808520	PELIC	BWI1	1
9/18/2011	608808520	PELIC	PHX3	1
9/18/2011	608808520	PELIC	PHX6	1
9/19/2011	608808520	PELIC	LEX1	3
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/19/2011	608808520	PELIC	LEX1	1
9/20/2011	608808520	PELIC	LEX1	2
9/20/2011	608808520	PELIC	PHX6	1
9/20/2011	608808520	PELIC	PHL1	1
9/20/2011	608808520	BTBRD	BTBR	1
9/20/2011	608808520	INN	IND1	1
9/20/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHL1	2
9/21/2011	608808520	PELIC	PHX6	2
9/21/2011	608808520	PELIC	LEX1	1
9/21/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	PELIC	PHX3	1
9/22/2011	608808520	ING	SEA6	1
9/22/2011	988123990	-1	-1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	PHL1	1
9/23/2011	608808520	PELIC	IND1	1
9/23/2011	608808520	PELIC	LEX1	1
9/24/2011	608808520	PELIC	PHX3	1
9/24/2011	608808520	PELIC	PHX6	1

Granularity

Oracle performs JOINs prior to the aggregation,
so it creates a pre-aggregation result set that looks like this. Explore how redshift performs the aggregation

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS	MERCHANT_CUSTOMER_ID_1	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	S	3	608808520	INJ	ABE2	1
9/18/2011	608808520	S	3	608808520	INN	LEX1	1
9/18/2011	608808520	S	3	608808520	INN	LEX1	1
9/18/2011	608808520	S	3	608808520	PELIC	PHX3	1
9/18/2011	608808520	S	3	608808520	BTS	ABE2	1
9/20/2011	988123990	S	1	608808520	INN	IND1	1
9/20/2011	608808520	S	6	608808520	INN	IND1	1
9/20/2011	988123990	S	1	608808520	PELIC	PHX6	1
9/20/2011	608808520	S	6	608808520	PELIC	LEX1	1
9/20/2011	988123990	S	1	608808520	PELIC	LEX1	1
9/20/2011	608808520	S	6	608808520	PELIC	LEX1	2
9/20/2011	608808520	S	6	608808520	PELIC	PHX6	1
9/20/2011	608808520	S	6	608808520	BTBRD	BTBR	1
9/20/2011	988123990	S	1	608808520	PELIC	PHL1	1
9/20/2011	608808520	S	6	608808520	PELIC	PHL1	1
9/20/2011	988123990	S	1	608808520	PELIC	LEX1	2
9/20/2011	988123990	S	1	608808520	BTBRD	BTBR	1
9/21/2011	608808520	1	2	608808520	PELIC	LEX1	1
9/21/2011	608808520	1	2	608808520	PELIC	PHL1	2
9/21/2011	608808520	1	2	608808520	PELIC	LEX1	1
9/21/2011	608808520	1	2	608808520	PELIC	PHX3	1
9/21/2011	608808520	S	2	608808520	PELIC	LEX1	1
9/21/2011	608808520	S	2	608808520	PELIC	PHX6	2
9/21/2011	608808520	S	2	608808520	PELIC	PHL1	2
9/21/2011	988123990	S	0	608808520	PELIC	PHL1	2
9/21/2011	988123990	S	0	608808520	PELIC	PHX3	1
9/21/2011	988123990	S	0	608808520	PELIC	PHX6	2
9/21/2011	988123990	S	0	608808520	PELIC	LEX1	1
9/21/2011	988123990	S	0	608808520	PELIC	LEX1	1
9/21/2011	608808520	S	2	608808520	PELIC	PHX3	1
9/21/2011	608808520	S	2	608808520	PELIC	LEX1	1
9/21/2011	608808520	1	2	608808520	PELIC	PHX6	2
9/22/2011	608808520	S	3	608808520	PELIC	PHX3	1
9/22/2011	608808520	S	3	608808520	ING	SEA6	1
9/22/2011	608808520	S	3	988123990	-1	-1	1
9/22/2011	988123990	S	0	988123990	-1	-1	1
9/23/2011	608808520	1	1	608808520	PELIC	IND1	1
9/23/2011	608808520	1	1	608808520	PELIC	PHL1	1
9/23/2011	988123990	S	0	608808520	PELIC	PHL1	1
9/23/2011	608808520	1	1	608808520	PELIC	IND1	1
9/24/2011	608808520	1	1	608808520	PELIC	PHX3	1
9/24/2011	608808520	1	1	608808520	PELIC	PHX6	1
9/24/2011	608808520	O	1	608808520	PELIC	PHX6	1
9/24/2011	608808520	S	0	608808520	PELIC	PHX6	1
9/24/2011	608808520	O	1	608808520	PELIC	PHX3	1
9/24/2011	608808520	S	0	608808520	PELIC	PHX3	1

Granularity

ACTIVITY_DAY	MERCHANT_CUSTOMER_ID	ORDER_METHOD	ORDER_UNITS	MERCHANT_CUSTOMER_ID_1	DISTRIBUTOR_ID	WAREHOUSE_ID	SHIP_UNITS
9/18/2011	608808520	S	3	608808520	INJ	ABE2	1
9/18/2011	608808520	S	3	608808520	INN	LEX1	1
9/18/2011	608808520	S	3	608808520	INN	LEX1	1
9/18/2011	608808520	S	3	608808520	PELIC	PHX3	1
9/18/2011	608808520	S	3	608808520	BTS	ABE2	1
9/18/2011	608808520	S	3	608808520	PELIC	BWI1	1
9/18/2011	608808520	S	3	608808520	PELIC	PHX6	1
9/19/2011	608808520	S	5	608808520	PELIC	LEX1	1
9/19/2011	608808520	S	5	608808520	PELIC	LEX1	1
9/19/2011	608808520	S	5	608808520	PELIC	LEX1	3
9/19/2011	608808520	S	5	608808520	PELIC	LEX1	1
9/20/2011	988123990	S	1	608808520	INN	IND1	1
9/20/2011	608808520	S	6	608808520	INN	IND1	1
9/20/2011	988123990	S	1	608808520	PELIC	PHX6	1
9/20/2011	608808520	S	6	608808520	PELIC	LEX1	1
9/20/2011	988123990	S	1	608808520	PELIC	LEX1	1
9/20/2011	608808520	S	6	608808520	PELIC	LEX1	2
9/20/2011	608808520	S	6	608808520	PELIC	PHX6	1
9/20/2011	608808520	S	6	608808520	BTBRD	BTBR	1
9/20/2011	988123990	S	1	608808520	PELIC	PHL1	1
9/20/2011	608808520	S	6	608808520	PELIC	PHL1	1
9/20/2011	988123990	S	1	608808520	PELIC	LEX1	2
9/20/2011	988123990	S	1	608808520	BTBRD	BTBR	1
9/21/2011	608808520	1	2	608808520	PELIC	LEX1	1
9/21/2011	608808520	1	2	608808520	PELIC	PHL1	2
9/21/2011	608808520	1	2	608808520	PELIC	LEX1	1
9/22/2011	608808520	S	3	608808520	PELIC	PHX3	1
9/22/2011	608808520	S	3	608808520	ING	SEA6	1
9/22/2011	608808520	S	3	988123990	-1	-1	1
9/22/2011	988123990	S	0	988123990	-1	-1	1
9/22/2011	988123990	S	0	608808520	ING	SEA6	1
9/22/2011	988123990	S	0	608808520	PELIC	PHX3	1
9/23/2011	988123990	S	0	608808520	PELIC	LEX1	1
9/23/2011	608808520	S	4	608808520	PELIC	LEX1	1
9/23/2011	608808520	S	4	608808520	PELIC	IND1	1
9/23/2011	608808520	S	4	608808520	PELIC	PHL1	1
9/23/2011	608808520	S	4	608808520	PELIC	IND1	1
9/23/2011	988123990	S	0	608808520	PELIC	IND1	1
9/23/2011	988123990	S	0	608808520	PELIC	IND1	1
9/23/2011	608808520	1	1	608808520	PELIC	LEX1	1
9/23/2011	608808520	1	1	608808520	PELIC	IND1	1
9/23/2011	608808520	1	1	608808520	PELIC	PHL1	1
9/23/2011	988123990	S	0	608808520	PELIC	PHL1	1
9/23/2011	608808520	1	1	608808520	PELIC	IND1	1
9/24/2011	608808520	1	1	608808520	PELIC	PHX3	1
9/24/2011	608808520	1	1	608808520	PELIC	PHX6	1
9/24/2011	608808520	O	1	608808520	PELIC	PHX6	1
9/24/2011	608808520	S	0	608808520	PELIC	PHX6	1
9/24/2011	608808520	O	1	608808520	PELIC	PHX3	1
9/24/2011	608808520	S	0	608808520	PELIC	PHX3	1

So we now know that these two tables have slightly different granularities and are more granular than simply ASIN and Day. We need to be aware of this and handle it correctly in our SQL to avoid over stating order and ship units

Granularity

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
;
```

```
SELECT
dds.ACTIVITY_DAY
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
dds.ACTIVITY_DAY
;
```

In the given example, previously we have seen that two independently run queries that we combined in Excel resulted in overstating both ORDER_UNITS and SHIP_UNITS in many cases

When you need to join tables with different granularities, and sum or count a column from one or both tables, subqueries can be used to produce data at the same granularities prior to joining

Granularity

```
(SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
) orders



(SELECT
dds.ACTIVITY_DAY
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
dds.ACTIVITY_DAY
) ships
```

Wrap the individual queries in parentheses and give them each an alias

Granularity

```
(SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
) orders



(SELECT
dds.ACTIVITY_DAY
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
dds.ACTIVITY_DAY
) ships
```

(indenting for ease of distinguishing sub-queries from outer queries)

Granularity

```
(SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
AND ddo.ASIN = '1589806808'
GROUP BY
ddo.ACTIVITY_DAY
) orders

(SELECT
dds.ACTIVITY_DAY
, SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
FROM D_DAILY_SHIPMENTS dds
WHERE dds.REGION_ID = 1
AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                         AND TO_DATE('20110924','YYYYMMDD')
AND dds.MARKETPLACE_ID = 1
AND dds.ASIN = '1589806808'
GROUP BY
dds.ACTIVITY_DAY
) ships

;
```

Granularity

```
SELECT
        (SELECT
        ddo.ACTIVITY_DAY
        , SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
        FROM D_DAILY_ORDERS ddo
        WHERE ddo.REGION_ID = 1
        AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                AND TO_DATE('20110924','YYYYMMDD')
        AND ddo.MARKETPLACE_ID = 1
        AND ddo.ASIN = '1589806808'
        GROUP BY
        ddo.ACTIVITY_DAY
        ) orders

        (SELECT
        dds.ACTIVITY_DAY
        , SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
        FROM D_DAILY_SHIPMENTS dds
        WHERE dds.REGION_ID = 1
        AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                AND TO_DATE('20110924','YYYYMMDD')
        AND dds.MARKETPLACE_ID = 1
        AND dds.ASIN = '1589806808'
        GROUP BY
        dds.ACTIVITY_DAY
        ) ships

;
```

Granularity

```
SELECT

FROM        (SELECT
            ddo.ACTIVITY_DAY
            , SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
            FROM D_DAILY_ORDERS ddo
            WHERE ddo.REGION_ID = 1
            AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND ddo.MARKETPLACE_ID = 1
            AND ddo.ASIN = '1589806808'
            GROUP BY
            ddo.ACTIVITY_DAY
            ) orders

            (SELECT
            dds.ACTIVITY_DAY
            , SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
            FROM D_DAILY_SHIPMENTS dds
            WHERE dds.REGION_ID = 1
            AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND dds.MARKETPLACE_ID = 1
            AND dds.ASIN = '1589806808'
            GROUP BY
            dds.ACTIVITY_DAY
            ) ships
;
```

Put those sub- queries in the FROM clause of an outer query

Granularity

```
SELECT

FROM        (SELECT
            ddo.ACTIVITY_DAY
            , SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
            FROM D_DAILY_ORDERS ddo
            WHERE ddo.REGION_ID = 1
            AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND ddo.MARKETPLACE_ID = 1
            AND ddo.ASIN = '1589806808'
            GROUP BY
            ddo.ACTIVITY_DAY
            ) orders
 JOIN
            (SELECT
            dds.ACTIVITY_DAY
            , SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
            FROM D_DAILY_SHIPMENTS dds
            WHERE dds.REGION_ID = 1
            AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND dds.MARKETPLACE_ID = 1
            AND dds.ASIN = '1589806808'
            GROUP BY
            dds.ACTIVITY_DAY
            ) ships

;
```

Granularity

```
SELECT
FROM        (SELECT
            ddo.ACTIVITY_DAY
            , SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
            FROM D_DAILY_ORDERS ddo
            WHERE ddo.REGION_ID = 1
            AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND ddo.MARKETPLACE_ID = 1
            AND ddo.ASIN = '1589806808'
            GROUP BY
            ddo.ACTIVITY_DAY
            ) orders
 JOIN
            (SELECT
            dds.ACTIVITY_DAY
            , SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
            FROM D_DAILY_SHIPMENTS dds
            WHERE dds.REGION_ID = 1
            AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND dds.MARKETPLACE_ID = 1
            AND dds.ASIN = '1589806808'
            GROUP BY
            dds.ACTIVITY_DAY
            ) ships
    ON orders.ACTIVITY_DAY = ships.ACTIVITY_DAY
;
```

Add a Join and Join Criteria

Granularity

```
SELECT

orders.ACTIVITY_DAY


FROM        (SELECT
            ddo.ACTIVITY_DAY
            , SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
            FROM D_DAILY_ORDERS ddo
            WHERE ddo.REGION_ID = 1
            AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND ddo.MARKETPLACE_ID = 1
            AND ddo.ASIN = '1589806808'
            GROUP BY
            ddo.ACTIVITY_DAY
            ) orders
 JOIN
            (SELECT
            dds.ACTIVITY_DAY
            , SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
            FROM D_DAILY_SHIPMENTS dds
            WHERE dds.REGION_ID = 1
            AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND dds.MARKETPLACE_ID = 1
            AND dds.ASIN = '1589806808'
            GROUP BY
            dds.ACTIVITY_DAY
            ) ships
    ON orders.ACTIVITY_DAY = ships.ACTIVITY_DAY
;
```

Add columns from either of the subqueries to the SELECT clause of the outer query

Granularity

Adding orders.ORDER_UNITS to SELECT Clause

```
SELECT
orders.ACTIVITY_DAY
, orders.ORDER_UNITS

FROM        (SELECT
            ddo.ACTIVITY_DAY
            , SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
            FROM D_DAILY_ORDERS ddo
            WHERE ddo.REGION_ID = 1
            AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                 AND TO_DATE('20110924','YYYYMMDD')
            AND ddo.MARKETPLACE_ID = 1
            AND ddo.ASIN = '1589806808'
            GROUP BY
            ddo.ACTIVITY_DAY
            ) orders
 JOIN
            (SELECT
            dds.ACTIVITY_DAY
            , SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
            FROM D_DAILY_SHIPMENTS dds
            WHERE dds.REGION_ID = 1
            AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                 AND TO_DATE('20110924','YYYYMMDD')
            AND dds.MARKETPLACE_ID = 1
            AND dds.ASIN = '1589806808'
            GROUP BY
            dds.ACTIVITY_DAY
            ) ships
    ON orders.ACTIVITY_DAY = ships.ACTIVITY_DAY
;
```

Granularity

Adding ships.SHIP_UNITS to SELECT Clause

```
SELECT
orders.ACTIVITY_DAY
, orders.ORDER_UNITS
, ships.SHIP_UNITS
FROM        (SELECT
            ddo.ACTIVITY_DAY
            , SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
            FROM D_DAILY_ORDERS ddo
            WHERE ddo.REGION_ID = 1
            AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND ddo.MARKETPLACE_ID = 1
            AND ddo.ASIN = '1589806808'
            GROUP BY
            ddo.ACTIVITY_DAY
            ) orders
 JOIN
            (SELECT
            dds.ACTIVITY_DAY
            , SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
            FROM D_DAILY_SHIPMENTS dds
            WHERE dds.REGION_ID = 1
            AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND dds.MARKETPLACE_ID = 1
            AND dds.ASIN = '1589806808'
            GROUP BY
            dds.ACTIVITY_DAY
            ) ships
    ON orders.ACTIVITY_DAY = ships.ACTIVITY_DAY
;
```

Granularity

```
SELECT
orders.ACTIVITY_DAY
, orders.ORDER_UNITS
, ships.SHIP_UNITS
FROM        (SELECT
            ddo.ACTIVITY_DAY
            , SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
            FROM D_DAILY_ORDERS ddo
            WHERE ddo.REGION_ID = 1
            AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND ddo.MARKETPLACE_ID = 1
            AND ddo.ASIN = '1589806808'
            GROUP BY
            ddo.ACTIVITY_DAY
            ) orders
 JOIN
            (SELECT
            dds.ACTIVITY_DAY
            , SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
            FROM D_DAILY_SHIPMENTS dds
            WHERE dds.REGION_ID = 1
            AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND dds.MARKETPLACE_ID = 1
            AND dds.ASIN = '1589806808'
            GROUP BY
            dds.ACTIVITY_DAY
            ) ships
    ON orders.ACTIVITY_DAY = ships.ACTIVITY_DAY
;
```

ACTIVITY_DAY	ORDER_UNITS
9/18/2011	3
9/19/2011	5
9/20/2011	7
9/21/2011	4
9/22/2011	3
9/23/2011	5
9/24/2011	2

Now each subquery produces a result set

ACTIVITY_DAY	SHIP_UNITS
9/18/2011	7
9/19/2011	6
9/20/2011	7
9/21/2011	7
9/22/2011	3
9/23/2011	4
9/24/2011	2

Granularity

And the results of the subqueries are joined, 1-to-1

ACTIVITY_DAY	ORDER_UNITS
9/18/2011	3
9/19/2011	5
9/20/2011	7
9/21/2011	4
9/22/2011	3
9/23/2011	5
9/24/2011	2

ACTIVITY_DAY	SHIP_UNITS
9/18/2011	7
9/19/2011	6
9/20/2011	7
9/21/2011	7
9/22/2011	3
9/23/2011	4
9/24/2011	2

Granularity

```
SELECT
orders.ACTIVITY_DAY
, orders.ORDER_UNITS
, ships.SHIP_UNITS
FROM        (SELECT
            ddo.ACTIVITY_DAY
            , SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
            FROM D_DAILY_ORDERS ddo
            WHERE ddo.REGION_ID = 1
            AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND ddo.MARKETPLACE_ID = 1
            AND ddo.ASIN = '1589806808'
            GROUP BY
            ddo.ACTIVITY_DAY
            ) orders
 JOIN
            (SELECT
            dds.ACTIVITY_DAY
            , SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
            FROM D_DAILY_SHIPMENTS dds
            WHERE dds.REGION_ID = 1
            AND dds.ACTIVITY_DAY BETWEEN TO_DATE('201109
                                     AND TO_DATE('201109
            AND dds.MARKETPLACE_ID = 1
            AND dds.ASIN = '1589806808'
            GROUP BY
            dds.ACTIVITY_DAY
            ) ships
    ON orders.ACTIVITY_DAY = ships.ACTIVITY_DAY
;
```

Producing the expected result

ACTIVITY_DAY	ORDER_UNITS	SHIP_UNITS
9/18/2011	3	7
9/19/2011	5	6
9/20/2011	7	7
9/21/2011	4	7
9/22/2011	3	3
9/23/2011	5	4
9/24/2011	2	2

Granularity

```
SELECT
orders.ACTIVITY_DAY
, orders.ORDER_UNITS
, ships.SHIP_UNITS
FROM        (SELECT
            ddo.ACTIVITY_DAY
            , SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
            FROM D_DAILY_ORDERS ddo
            WHERE ddo.REGION_ID = 1
            AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND ddo.MARKETPLACE_ID = 1
            AND ddo.ASIN = '1589806808'
            GROUP BY
            ddo.ACTIVITY_DAY
            ) orders
 JOIN
            (SELECT
            dds.ACTIVITY_DAY
            , SUM(dds.SHIPPED_UNITS) as SHIP_UNITS
            FROM D_DAILY_SHIPMENTS dds
            WHERE dds.REGION_ID = 1
            AND dds.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
                                     AND TO_DATE('20110924','YYYYMMDD')
            AND dds.MARKETPLACE_ID = 1
            AND dds.ASIN = '1589806808'
            GROUP BY
            dds.ACTIVITY_DAY
            ) ships
    ON orders.ACTIVITY_DAY = ships.ACTIVITY_DAY
;
```

ACTIVITY_DAY	ORDER_UNITS	SHIP_UNITS
9/18/2011	3	7
9/19/2011	5	6
9/20/2011	7	7
9/21/2011	4	7
9/22/2011	3	3
9/23/2011	5	4
9/24/2011	2	2

Producing the expected result

ACTIVITY_DAY	ORDER_UNITS	SHIP_UNITS
9/18/2011	3	7
9/19/2011	5	6
9/20/2011	7	7
9/21/2011	4	7
9/22/2011	3	3
9/23/2011	5	4
9/24/2011	2	2

WITH CLAUSE

WITH CLAUSE

- The SQL WITH clause allows you to give a sub-query block a name (a process also called sub-query refactoring), which can be referenced in several places within the main SQL query.
- The clause is used for defining a temporary relation such that the output of this temporary relation is available and is used by the query that is associated with the WITH clause.
- Queries that have an associated WITH clause can also be written using nested sub-queries but doing so add more complexity to read/debug the SQL query.
- WITH clause is not supported by all database system.
- The name assigned to the sub-query is treated as though it was an inline view or table

WITH CLAUSE

Syntax:

```
WITH query_name (column_name1, ...) AS
    (SELECT ...)

SELECT ...
```

ETL – More Wildcards

More Wildcards

- We've previously talked about two wildcards available for use in ETL queries:
  - Job Profile ID Wildcard {JOB_PROFILE_ID}
  - Run Date Wildcard {RUN_DATE_YYYYMMDD}

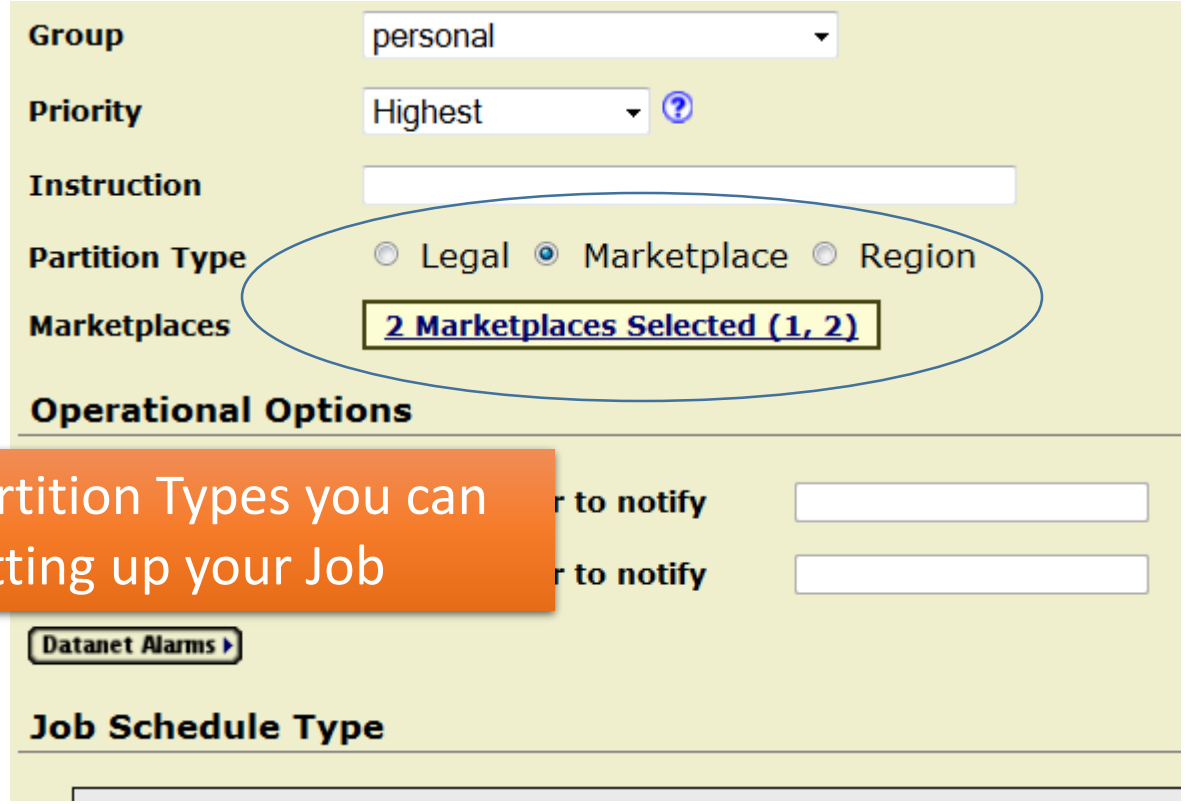
- There are also wildcards that use information you specify in the Job section of your query to input values

More Wildcards

- Region ID {REGION_ID}
- Region code {REGION_CODE}
- Marketplace ID {MARKETPLACE_ID}
- Legal Entity ID {LEGAL_ENTITY_ID}
- Free Form {FREE_FORM}

More Wildcards

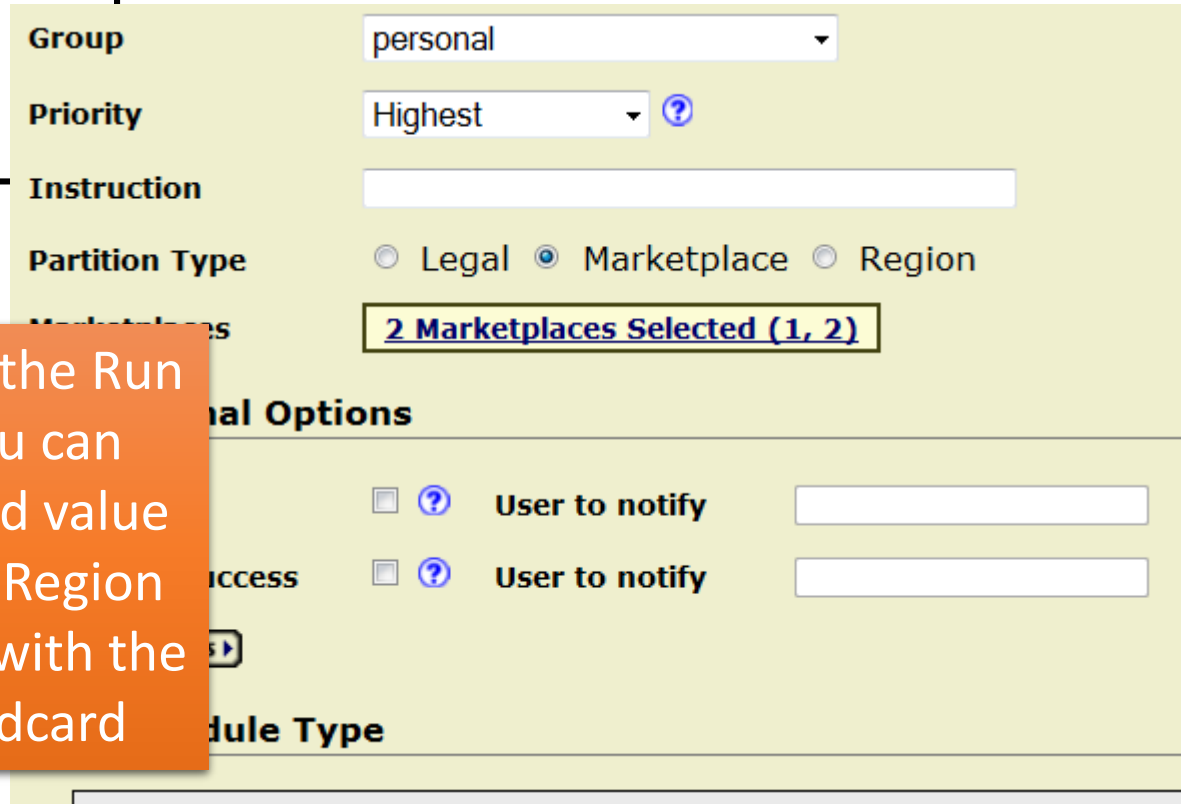
- Region ID {REGION_ID}
- Marketplace ID {MARKETPLACE_ID}
- Legal Entity ID {LEGAL_ENTITY_ID}



Notice the three Partition Types you can select when setting up your Job

More Wildcards

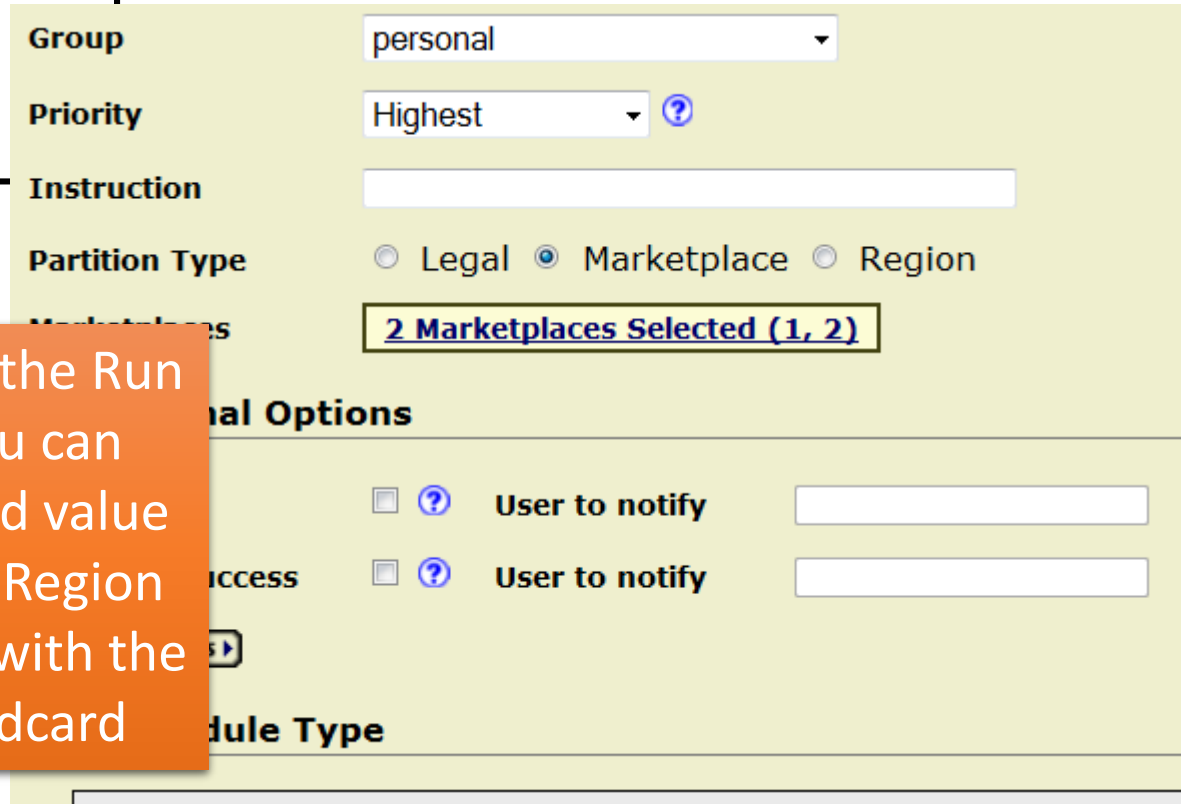
```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
   AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
GROUP BY
ddo.ACTIVITY_DAY
;
```



Just as you can with the Run Date Wildcard, you can replace a hard-coded value for Marketplace ID, Region ID or Legal Entity ID with the corresponding wildcard

More Wildcards

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
   AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = {MARKETPLACE_ID}
GROUP BY
ddo.ACTIVITY_DAY
;
```

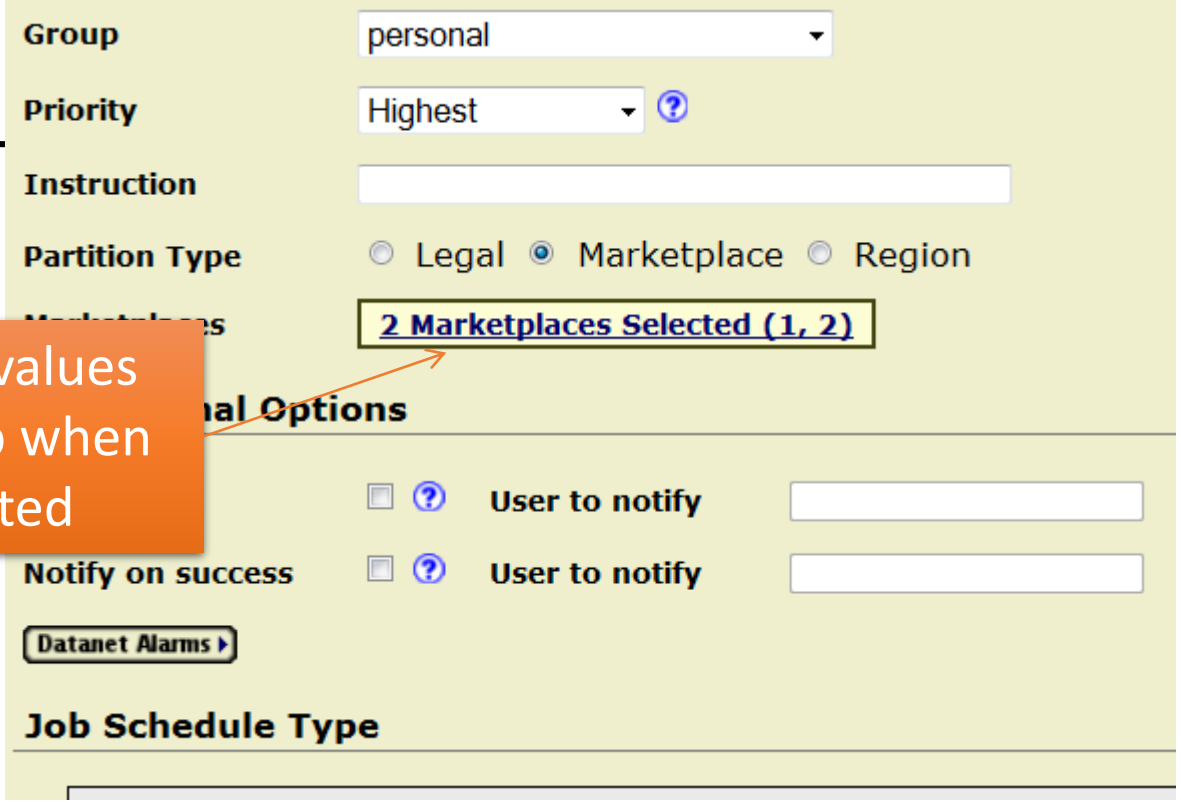


Just as you can with the Run Date Wildcard, you can replace a hard-coded value for Marketplace ID, Region ID or Legal Entity ID with the corresponding wildcard

More Wildcards

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
   AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = {MARKETPLACE_ID}
GROUP BY
ddo.ACTIVITY_DAY
;
```

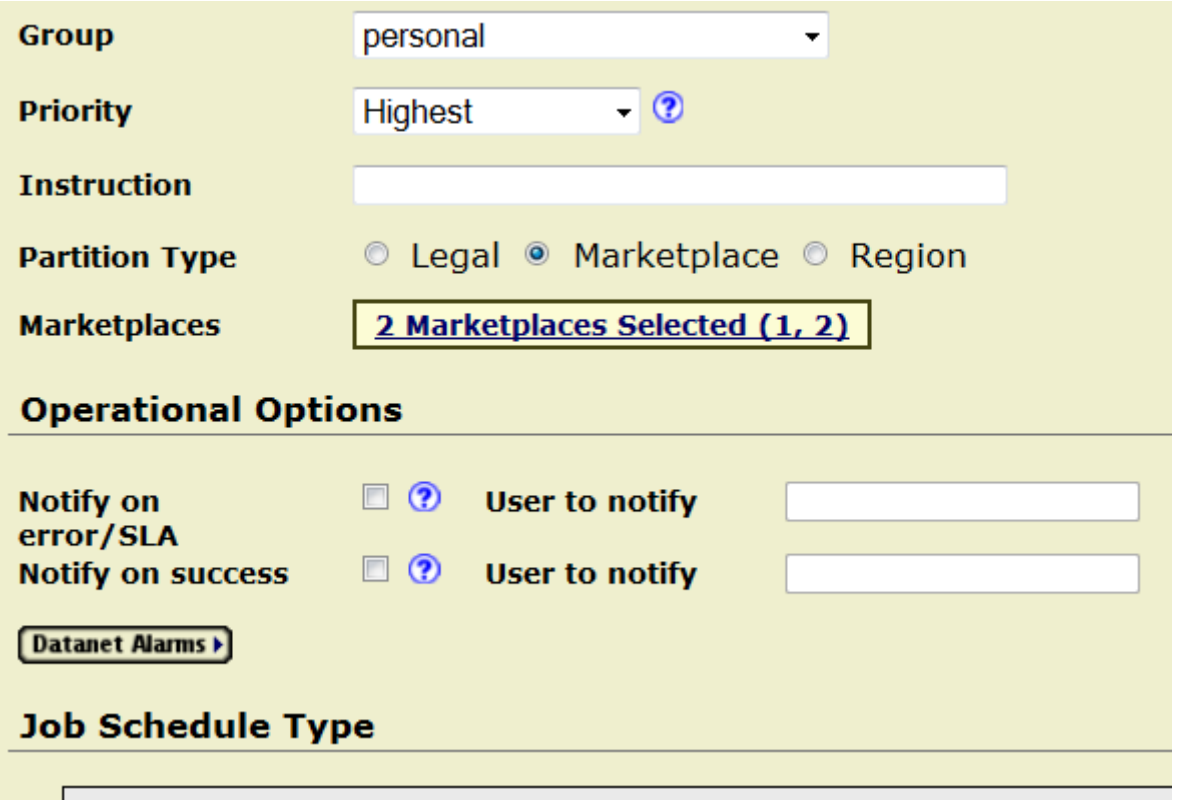
And it will use the values selected in your Job when the SQL is executed



More Wildcards

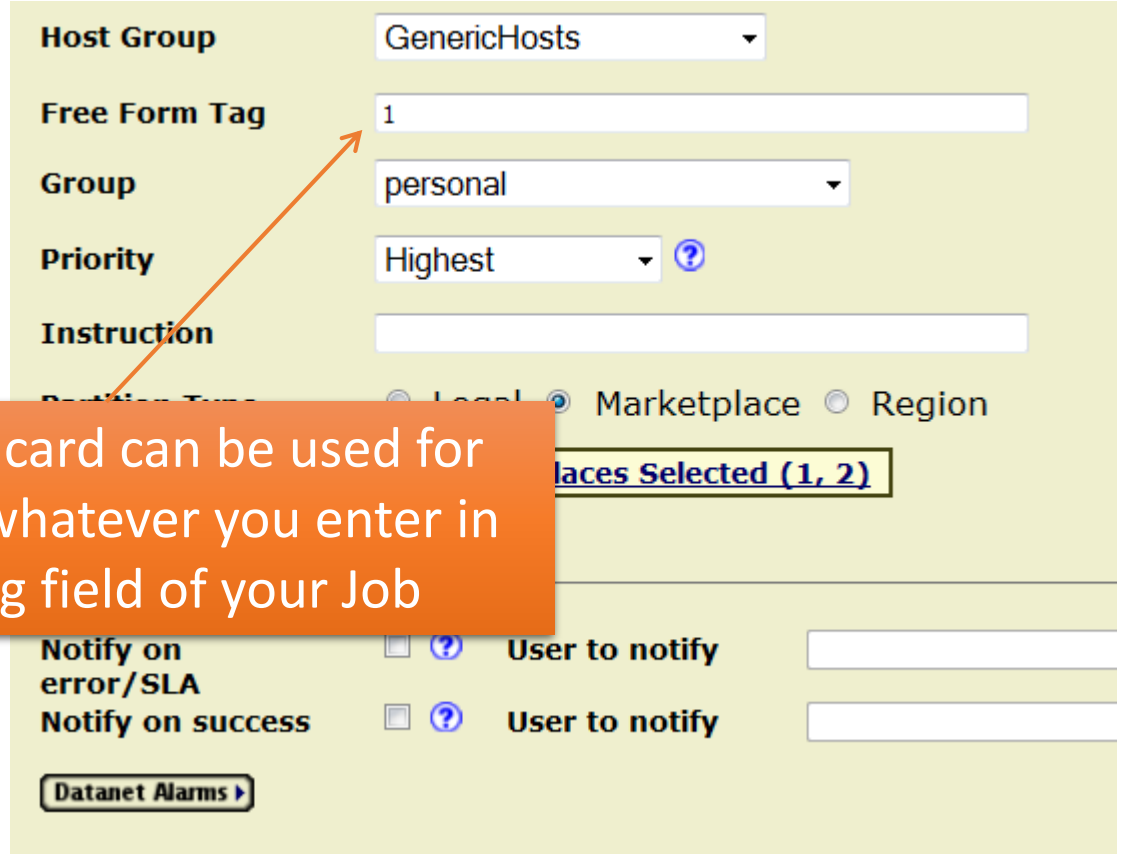
```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
   AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = {MARKETPLACE_ID}
GROUP BY
ddo.ACTIVITY_DAY
;
```

Since you can only select one Partition Type, you can only use one of these three wildcards – the one corresponding to the partition type you selected. Hence, in this case Market place is the wildcard used in the query



More Wildcards

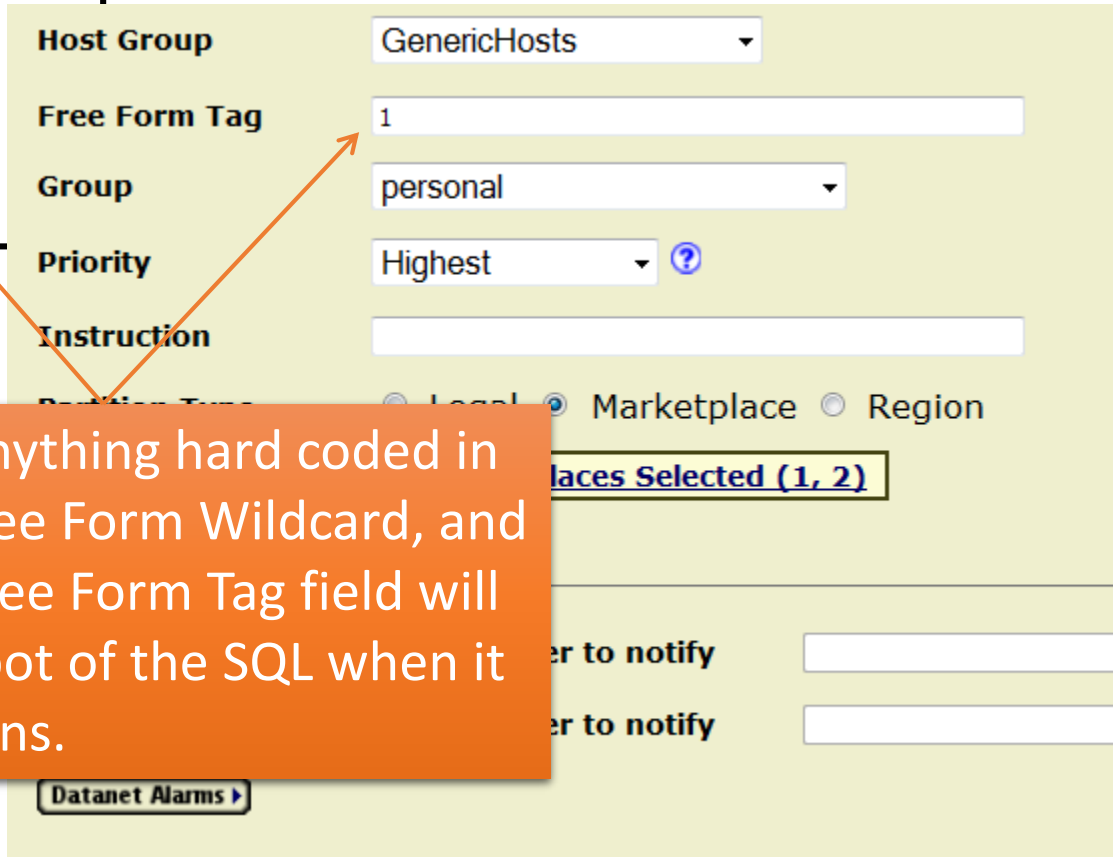
- Free Form {FREE_FORM}



The Free Form Wildcard can be used for anything, and uses whatever you enter in the Free Form Tag field of your Job

More Wildcards

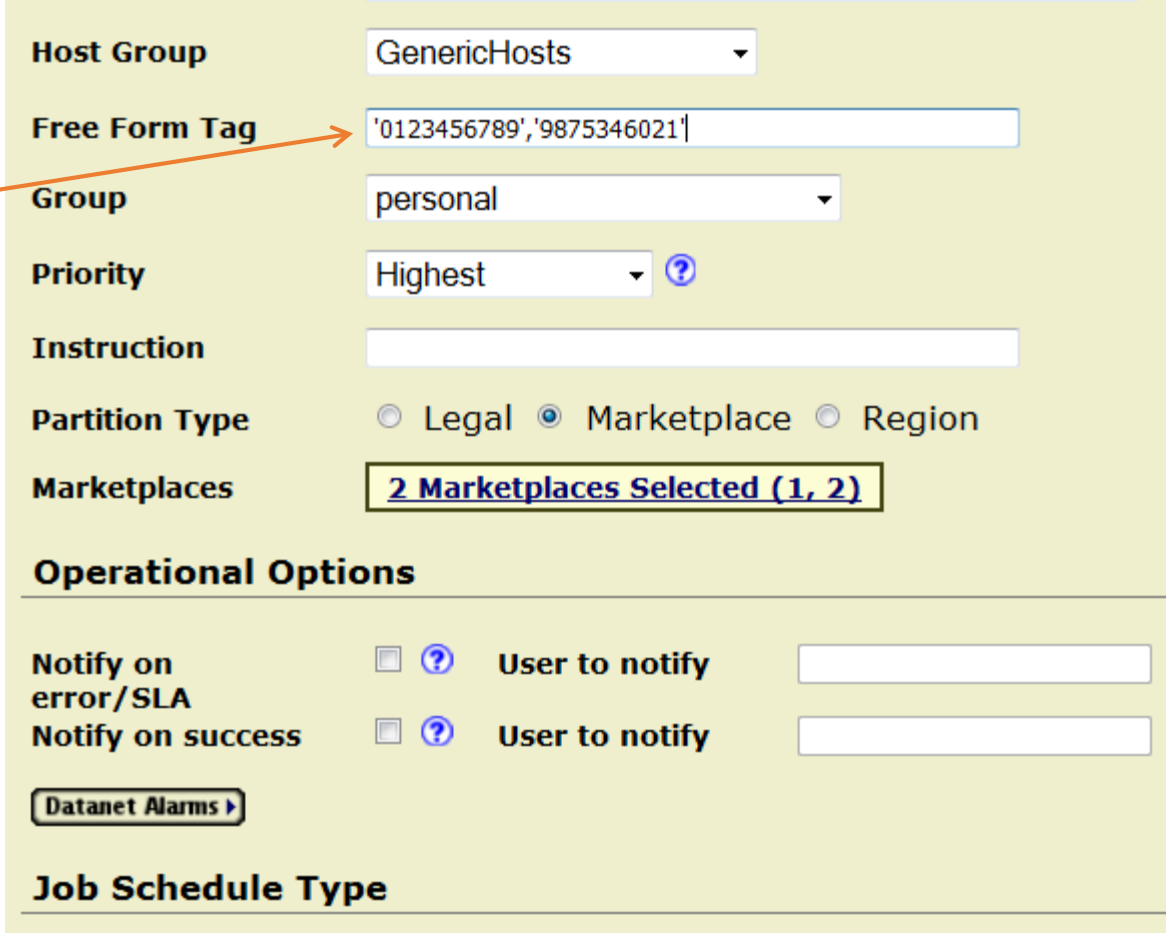
```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = {FREE_FORM}
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
   AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = {MARKETPLACE_ID}
GROUP BY
ddo.ACTIVITY_DAY
;
```



So we can replace anything hard coded in the query with the Free Form Wildcard, and whatever is in the Free Form Tag field will be entered in that spot of the SQL when it runs.

More Wildcards

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY BETWEEN TO_DATE('20110918','YYYYMMDD')
   AND TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = {MARKETPLACE_ID}
AND ddo.ASIN IN ({FREE_FORM})
GROUP BY
ddo.ACTIVITY_DAY
;
```



Yielding lots of possibilities

More Wildcards

Extract Job Profile - dgarling TESTING 4

Created by	Dan Garlington
Updated by	Dan Garlington
Valid from	2011-09-28 12:42:21 PDT
Revision	7 (log) (history)
File template	

Help | Edit Profile | Delete Profile and ALL Jobs | Copy Profile with NO Jobs

Publishers applied to this Profile

Help | New

	Publisher Type	Format	Values
Edit Delete	Email	Text	{FREE_FORM}@amazon.com

Main Template (Default) | New Job with 1-Click | Create 1-Click Template

The above Job Profile is being used by these jobs

Select All Jobs | Run Jobs | Edit Jobs | Delete Jobs | Copy Jobs | New | View Deleted Jobs | Help

Job #2060083 Revision #3: dgarling TESTING 4

	Page	SLA	Parallel	Scheduled	User	Group	Logical DB
Run | Edit | Delete | Copy | SDL | History | Datanet Alarms		0	1	Not scheduled	RETAIL_RO	personal	L-DW

Marketplaces: 1, 2	**Warehouse:**	**Free form tag:** dgarling	**Priority:** Highest	**Timezone:** America/Los_Angeles
Host Group: GenericHosts				

The above Job Profile is being used by the above jobs

Select All Jobs | Run Jobs | Edit Jobs | Delete Jobs | Copy Jobs | New | View Deleted Jobs | Help

You can even use the Free Form Wildcard in your publishing path or Email addresses

More Wildcards

Publishers applied to this Profile

Help | New | Edit | Delete

Publisher Type	Format	Values
Email	Text	{FREE_FORM}@amazon.com

Main Template (Default) | New Job with 1-Click | Create 1-Click Template

The above Job Profile is being used by these jobs

New | View Deleted Jobs | Help

Select All Jobs | Run Jobs | Edit Jobs | Delete Jobs | Copy Jobs

Job #2060083 Revision #3: dgarling TESTING 4

	Page	SLA	Parallel	Scheduled	User	Group	Logical DB
Run | Edit | Delete | Copy | SDL | History | Datanet Alarms		0	1	Not scheduled	RETAIL_RO	personal	L-DW

Marketplaces:	Warehouse:	Free form tag:	Priority:	Timezone:
1, 2		dgarling	Highest	America/Los_Angeles

Host Group: GenericHosts

Job #2086084 Revision #1: Job for profile dgarling TESTING 4b

	Page	SLA	Parallel	Scheduled	User	Group	Logical DB
Run | Edit | Delete | Copy | SDL | History | Datanet Alarms		0	1	Not scheduled	APPS_RO	personal	L-DW

Marketplaces:	Warehouse:	Free form tag:	Priority:	Timezone:
3240		larrych	Highest	Asia/Shanghai

Host Group: GenericHosts

The above Job Profile is being used by the above jobs

New | View Deleted Jobs | Help

The Power of Wildcards becomes more apparent if you have multiple jobs in a single query.

More Wildcards

JUST LIKE THE RUNDATE WILDCARD…

ONLY WHEN YOUR QUERY INCLUDES A WILDCARD DOES THE VALUE YOU SELECT IMPACT THE RESULTS

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = {REGION_ID}
AND ddo.ACTIVITY_DAY =
TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
GROUP BY
ddo.ACTIVITY_DAY
;
```

WILDCARD = IMPACTED

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY =
TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
GROUP BY
ddo.ACTIVITY_DAY
;
```

More Wildcards

JUST LIKE THE RUNDATE WILDCARD…

ONLY WHEN YOUR QUERY INCLUDES A WILDCARD DOES THE VALUE YOU SELECT IMPACT THE RESULTS

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = {REGION_ID}
AND ddo.ACTIVITY_DAY =
TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
GROUP BY
ddo.ACTIVITY_DAY
;
```

WILDCARD = IMPACTED

```
SELECT
ddo.ACTIVITY_DAY
, SUM(ddo.ORDERED_UNITS) as ORDER_UNITS
FROM D_DAILY_ORDERS ddo
WHERE ddo.REGION_ID = 1
AND ddo.ACTIVITY_DAY =
TO_DATE('20110924','YYYYMMDD')
AND ddo.MARKETPLACE_ID = 1
GROUP BY
ddo.ACTIVITY_DAY
;
```

NO WILDCARD = NOT IMPACTED

- Using other Operators

➢ETL Topics

1) Explain Plan
2) Troubleshooting
3) Altering Job Runs
4) Job Performance History
5) Help

Reference Link: https://w.amazon.com/index.php/DanGSQLClass/IntroToSqlEtl/Lesson2#HTheWHEREClause2013SQL2019sFilter

Lesson 6 Assignment

As always, remember to include WHERE clause conditions on any and all Partitioned columns, and run any query you write through Explain Plan before running it.

1. Create a segment of the following ASINs: 0394873742, 037584726X, 0789399903, 0345431391, 0375847278, 037584726X, 0887767702
2. Create a query that emails you the list of distinct ASINs in this segment.
3. Use the query you created as a subquery in a query to find a list of all POs that were placed on 4/6/2009 in the US that included those ASINs. In your results, include the PO, Vendor Code, ASIN, and quantity confirmed. (Hint: check out D_DISTRIBUTOR_ORDER_ITEMS. The column QUANTITY_ORDERED indicates the number of units confirmed.)
4. Edit the query to switch out the condition on Legal Entity ID in your WHERE clause to use the Legal Entity ID wildcard, and rerun the query. Make sure your Job is set up to be partitioned by Legal Entity ID.
5. Edit the query to switch out the segment ID in your subquery for the Free Form Tag Wildcard, and edit your job to put the segment ID in the Free Form Tag field.
6. Edit the query to add a JOIN to the VENDORS table to get the name of each Vendor.
7. Edit the query to add a JOIN to the D_MP_ASINS_ESSENTIALS table to get the title of each ASIN.
8. Edit the query to remove the PO Field, and sum the number of units confirmed per ASIN, per Vendor.
9. Edit the query to return only those ASINs where the sum of units ordered was greater than 3.

END