DATA ANALYTIC SERVICES - TRAINING

Querying a Relational Database

UNION
JOIN 2 Tables
JOIN Multiple Tables

Matthew Morris
Git: DasTraining
Msmorris@costco.com

PREVIOUSLY IN DATA ANALYTICS

FILTERS = , !=, >, <
IN, NOT IN, BETWEEN, LIKE, NOT LIKE
SUM, MIN, MAX, COUNT
GROUP BY, HAVING
COMMENTING

Matthew Morris
Git: Morrisdata
Msmorris@costco.com

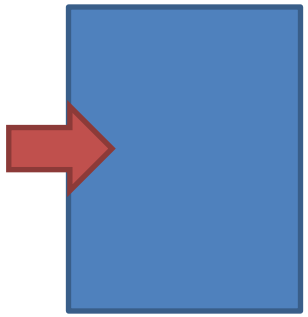
Querying a relational Database

SELECT
FROM
JOIN
ON
WHERE
GROUP BY
HAVING
UNION
ORDER BY
LIMIT

UNION

Unions

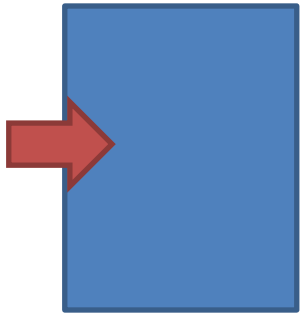
FY17



```
SELECT fy, pd, store_name, week1,
week2, week3 week4
FROM  FY17
```

UNION

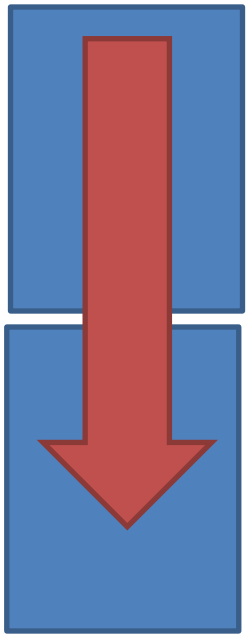
FY18



```
SELECT fy, pd, store_name, week1,
week2, week3 week4
FROM  FY18
```

Unions

FY17



FY18

```
SELECT fy, pd, store_name, week1,
week2, week3 week4
FROM  FY17
UNION
SELECT fy, pd, store_name, week1,
week2, week3 week4
FROM  FY18
```

COLUMNS
CONDITIONS
UNION and UNION ALL
ORDER BY

JOIN 1 Table

Joins

Querying a Relational Database

Joins

Query a Relational Database

LEFT/PRIMARY

RIGHT/SECONDARY

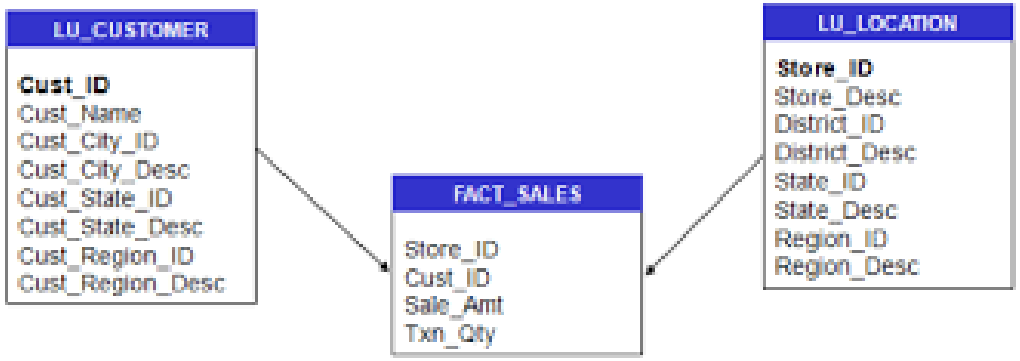
What table is the transaction table?

If you wanted to link on the lowest level of detail to the other tables what fields would you use?

Joins

	SALES		
	FIELD	TYPE	LENGTH
	ID	PK	1
	ARTIST	Char	25
	SONG	Char	225
	ALBUM	Char	225

Create a rough sketch with how INWITMP would join to: INDDESP, INWCTLP, INITMMP



Understanding data:

Run counts on the links

Answer why counts or data is different

Read documentation
or speak with a subject matter expert?

Joins

a = INWITMP
b = INWCTLP

```
SELECT a.WICMPY, a.WIWHS5, b.WCCMPY, b.WCWHS5

FROM RPT_MOD.VW_INPRDINV_INWITMP a

-- company link
JOIN RPT_MOD.VW_INPRDINV_INWCTLP b
ON a.WICMPY = b.WCCMPY

-- warehouse link
JOIN RPT_MOD.VW_INPRDINV_INWCTLP b
ON a.WIWHS5= b.WCWHS5
```

Joins

1. Create separate queries to join each table to RPT_MOD.VW_INPRDINV_INWITMP
    a. INWCTLP to Sales
    b. INDDESP to Sales
    c. INITMMP to Sales
2. Use this as an opportunity to bring fields in from both tables.
3. Try out some aggregations or Wild card searches. Stretch with an Aggregate and a Group by

Joins

```
SELECT c.field, a.field, b.field, a.field,c.field

FROM table1 a

JOIN table2 b
ON a.field=b.field

JOIN table3 c
ON a.field=c.field
```

LEFT/PRIMARY

RIGHT/SECONDARY

Example

Employees

id	first_name	last_name
2	Gabe	Moore
3	Doreen	Mandeville
5	Simone	MacDonald
7	Madisen	Flateman
11	Ian	Paasche
13	Mimi	St. Felix

Salaries

id	current_salary
2	50000
3	60000
7	55000
11	75000
13	120000
17	70000

Inner Join

Employees

id	first_name	last_name
2	Gabe	Moore
3	Doreen	Mandeville
5	Simone	MacDonald
7	Madisen	Flateman
11	Ian	Paasche
13	Mimi	St. Felix

Salaries

id	current_salary
2	50000
3	60000
7	55000
11	75000
13	120000
17	70000

```
FROM Employees a
JOIN Salaries b
ON a.id=b.id
```

id	first_name	last_name	id	current_salary
2	Gabe	Moore	2	50000
3	Doreen	Mandeville	3	60000
7	Madisen	Flateman	7	55000
11	Ian	Paasche	11	75000
13	Mimi	St. Felix	13	7000

Left Outer Join

Employees

id	first_name	last_name
2	Gabe	Moore
3	Doreen	Mandeville
5	Simone	MacDonald
7	Madisen	Flateman
11	Ian	Paasche
13	Mimi	St. Felix

Salaries

id	current_salary
2	50000
3	60000
7	55000
11	75000
13	120000
17	70000

FROM Employees a
LEFT JOIN Salaries b
ON a.id=b.id

id	first_name	last_name	id	current_salary
2	Gabe	Moore	2	50000
3	Doreen	Mandeville	3	60000
5	Simone	MacDonald	NULL	NULL
7	Madisen	Flateman	7	55000
11	Ian	Paasche	11	75000
13	Mimi	St. Felix	13	120000

Right-Outer Join

Employees

id	first_name	last_name
2	Gabe	Moore
3	Doreen	Mandeville
5	Simone	MacDonald
7	Madisen	Flateman
11	Ian	Paasche
13	Mimi	St. Felix

Salaries

id	current_salary
2	50000
3	60000
7	55000
11	75000
13	120000
17	70000

```
FROM Employees a
RIGHT JOIN Salaries b
ON a.id=b.id
```

id	first_name	last_name	id	current_salary
2	Gabe	Moore	2	50000
3	Doreen	Mandeville	3	60000
7	Madisen	Flateman	7	55000
11	Ian	Paasche	11	75000
13	Mimi	St. Felix	13	120000
NULL	NULL	NULL	17	70000

Left Exception Join

Employees

id	first_name	last_name
2	Gabe	Moore
3	Doreen	Mandeville
5	Simone	MacDonald
7	Madisen	Flateman
11	Ian	Paasche
13	Mimi	St. Felix

Salaries

id	current_salary
2	50000
3	60000
7	55000
11	75000
13	120000
17	70000

```
FROM Employees a
LEFT JOIN Salaries b
ON a.id=b.id
WHERE b.id IS NULL
```

id	first_name	last_name	id	current_salary
5	Simone	MacDonald	NULL	NULL

Right Exception Join

Employees

id	first_name	last_name
2	Gabe	Moore
3	Doreen	Mandeville
5	Simone	MacDonald
7	Madisen	Flateman
11	Ian	Paasche
13	Mimi	St. Felix

Salaries

id	current_salary
2	50000
3	60000
7	55000
11	75000
13	120000
17	70000

```
FROM Employees a
RIGHT JOIN Salaries b
ON a.id=b.id
WHERE a.id IS NULL
```

id	first_name	last_name	id	current_salary
NULL	NULL	NULL	17	70000

Cross Join

Employees

id	first_name	last_name
2	Gabe	Moore
3	Doreen	Mandeville
5	Simone	MacDonald
7	Madisen	Flateman
11	Ian	Paasche
13	Mimi	St. Felix

Salaries

id	current_salary
2	50000
3	60000
7	55000
11	75000
13	120000
17	70000

FROM Employees a
CROSS JOIN Salaries b
ON a.id=b.id

id	first_name	last_name	id	current_salary
2	Gabe	Moore	2	50000
3	Doreen	Mandeville	3	60000
5	Simone	MacDonald	NULL	NULL
7	Madisen	Flateman	7	55000
11	Ian	Paasche	11	75000
13	Mimi	St. Felix	13	120000
NULL	NULL	NULL	17	70000

Joins

Types of Joins

Inner Join	Match in both tables
Left-Outer Join	Includes data from the primary table that may not have matches
Right-Outer Join	Includes data from the secondary table that may not have matches
Exception Join	Returns Primary table data that does not match with the secondary table
Right-Exception Join	Returns Secondary table data that does not match with the Primary table
Cross Join	Returns all data whether a match exists or not

Q & A

“Sometimes questions are more important than answers.”
– Nancy Willard

Conclusion

Find the lowest level detail between tables.

Place the transactional or largest table on the left

FEEDBACK

CLASS : BASIC JOINS

QUESTION:

What is the benefit using ON for join rather than WHERE?