Hands on Session Tableau Prep & Desktop

Optional: Advanced SQL Queries

Abid Hussain,
Associate Professor
Center for Business Data Analytics
Department of Digitalization, Copenhagen Business School, Copenhagen, Denmark.

Contact: ah.digi@cbs.dk

AGENDA

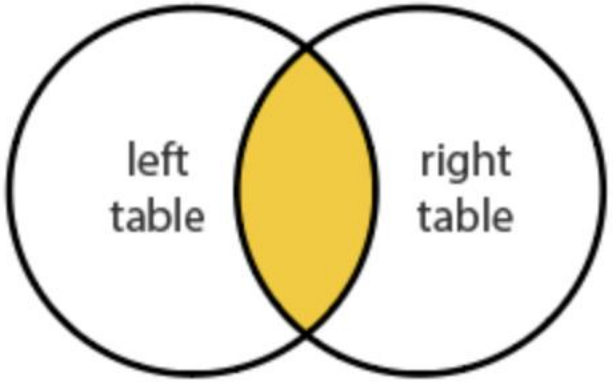
Agenda

- Tableau Prep for data cleaning
- Recorded training for Tableau (at the end or after lecture)
- Advanced topics for Structured Query Language (SQL) (Optional)
  - Left Join in detailed
  - Analytical functions

SQL QUERIES ADVANCED

JOINS VS LEFT JOINS

INNER JOIN



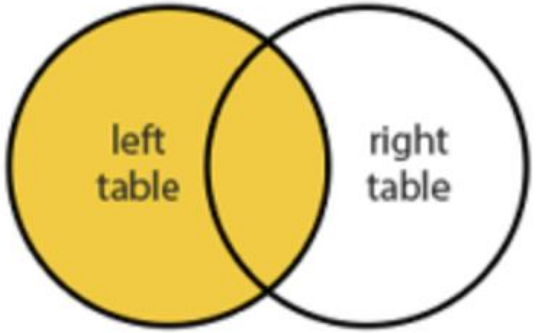
- SELECTS records with the matching values in both tables
  - **Example:**

    select * **from** *student* **join** *takes* **on** *student*.*ID* = *takes*.*ID*;

    or

    S**elect** * **from** *student*, *takes* **where** *student*.*ID* = *takes*.*ID*;

LEFT (OUTER) JOIN



- The **left outer join** preserves tuples only in the relation named before (to the left of) the **left outer join** operation

Example: Get a list of student who did not take a course

```
select *
from student s left join takes t
on s.id=t.id where t.id is null;
```

Left Join / Union and Join putting together

- Left join is useful to find missing membership
  - Find Students who didn't take any course
  - Find Teachers who didn't supervise in a given year
  - Find a product that didn't sell at all (or in a given timeframe)
- Let us try to get following report from the database
  - Find Those who didn't take any course
  - Find Those who didn't get any Supervision
  - Union them and use Case

```
Select st.id,
case when nosubject.sid is not null then 'X' end NoSubject,
case when noSupervision.sid is not null then 'X' end NoSupervision
from
(select s.id from student s left join takes t on
s.id = t.id
where t.id is null
UNION
select s.id from student s left join advisor a on
s.id = a.s_id
where a.s_id is null) st left join
(select s.id sid, t.id from student s left join takes t on
s.id = t.id
where t.id is null) nosubject
on st.id = nosubject.sid
Left join
(select s.id sid, a.s_id from student s left join advisor a on
s.id = a.s_id
where a.s_id is null) noSupervision
on st.id = noSupervision.sid
```

Student Id	Name	No Course Taken	No Supervision Taken
1001	Harvey Specter		X
1002	Michael Ross	X	
1003	Louiss Litt	X	X

Advanced (Analytical) Functions

- Analytical functions are powerful tool to build powerful reports very fast
  - The syntax can seem complicated
  - A general syntax :

    AnalyticalFuncName([arguments])

    OVER (

    [PARTITION BY partition_expression,…]

    [ORDER BY sort_expression, … [ASC|DESC]])
  - Examples:
    - Find Top 5 Stores by Sale
    - Find top Nth product with respect to sale in particular region
    - Compare Total Sale Per month with immediate previous month

Advanced (Analytical) Functions

- Analytical functions
  - RANK() => Assign rank to a row based on column
  - FIRST_VALUE => Find first value within a group or dataset
  - NTH_VALUE => Find nth value within a group or dataset
  - LAG() => Display values after the occurance of current row
  - LEAD() => Display values before the occurance of current row

Advanced (Analytical) Functions – RANK()

- Assign rank to a row based on column
- Examples
  - Using ShoeX Database. Generate a report showing the top store in each country with total sale.

```
select * from
(
  select si.storeid, storename, country, sum(finalprice) totalsale,
      Rank() over(partition by country order by sum(finalprice) desc) SaleRank
from saletransaction t join storeinfo si
on t.storeid = si.storeid join saleitem i on t.transactionid = i.transactionid
group by si.storeid, storename, country) T

where salerank = 1
```

Advanced (Analytical) Functions – FIRST_VALUE()

- Retrieve First value within a group or dataset
- Examples
  - Using ShoeX Database. Generate a report showing most popular product sold in each country with total sale value and comparing other products sale difference to the top product.

```
select H.categoryid, categoryname, subcategoryname, country, totalsale, TopSale, TotalSale-topsale DiffToTopProduct from
(select categoryid, country, totalsale,
 first_Value(totalsale) over(partition by country order by totalsale desc) TopSale
 From
 (
Select categoryid, country, sum(finalprice) totalsale
from saletransaction t join storeinfo si on
t.storeid = si.storeid join saleitem i on
t.transactionid = i.transactionid
group by categoryid, country
) T
)H,
(select categorymapid categoryid, categoryname, subcategoryname from categorymap cm, maincategory mc, subcategory sc
where cm.maincategoryid = mc.categoryid and cm.subcategoryid = sc.subcategoryid) cat
where H.categoryid = cat.categoryid
```

Advanced (Analytical) Functions – LAG()

- Display value before the occurance of current row
- Examples
  - Using ShoeX Database. Generate a report showing total sale each month and compare it with previous month.
  - Notice 2 digit formatting and cast

```
select country, monthofsale, totalsale,
Lag(totalsale) OVER(partition by country order by monthofsale) PreviousMonthSale
from
(Select country, cast(concat(date_part('year', purchasedate), To_Char(date_part('month', purchasedate), 'fm00')) as int) monthofsale ,
sum(finalprice) totalsale
from saletransaction t join storeinfo si on
t.storeid = si.storeid join saleitem i on
t.transactionid = i.transactionid
group by country, cast(concat(date_part('year', purchasedate), To_Char(date_part('month', purchasedate), 'fm00')) as int)) T
order by country, monthofsale
```

THANK YOU