# Applying Analytical Patterns

**Day 2 Lecture**
*Data Engineering Design Patterns at Meta - Funnel Accounting*

**Transcript:**

0:00
funnels are a massive part of analytics and data engineering and what does that mean so you have a funnel is something
0:05
where you have a lot of people at the top and you have not as many people at the bottom and so you want to push people through funnels to make more
0:12
money to get more growth to get more engagement there's so many different paths that funnels are necessary for and
0:18
the data engineering behind it is somewhat complicated in this course we're going to be covering how to do
0:23
good funnel analysis and uh I hope you enjoy it because we're going to do funnel analysis just like they do at
0:28
meta
0:34
so I'm sure y'all have seen this meme uh this meme is uh I don't know I've probably seen this meme at least like
0:40
once a month on LinkedIn where the the technical interview for data engineering
0:45
the SQL interview they're going to ask you all sorts of crazy stuff and you might get like these crazy like
0:52
self-join questions or ranking questions or like and then like and then when you actually do the job like the odds that
0:59
you actually use those functions or a little bit lower I found that like that
1:04
there's more overlap between the data engineering SQL interview and the data engineering job than there is between
1:10
like the software engineering leak code interview and the software engineering job so in that way like I can't complain
1:15

too much about the data engineering SQL interview but it's you know like there's definitely still like they they they try
1:22
to test you try to grill you pretty hard so things you'll see in the data
1:28
engineering SQL interview that you'll almost never do on the job so um I've been in this situation a
1:36
couple times where I've been interviewing at companies and I will come up with a solution that uses a
1:41
window function and then they ask me to rewrite it without using a window function
1:47
and like I don't know man like what like i' I've been there like more than once in
1:53
my career and when they ask me to do that I'm always like makes me want to like throw a chair because it's like I
2:00
solved the problem bro like like why are you making me solve the problem again and some companies have this like absurd
2:09
obsession with like an SQL but not even like anql because a lot of window functions are an SQL now but like more
2:16
like 2003 ansy SQL they're all about that right and in this case though like
2:23
when if you were trying to solve a problem with a window function and you
2:28
do want to convert it to a problem that is not using a window function uh Sometimes It's Tricky can be actually a
2:35
very tricky problem especially like with like ranking and stuff like that some it can actually get really gnarly but at
2:41
least for the case of um like uh a lot of times you use like lead and lag and
2:48
lead and lag can easily be uh replicated with a self join where you do a self
2:54
join on like date plus one like date equals date plus one and then you get the two two records on the same row and
3:01
then you can essentially do that that's what they're usually looking for when they're asking you to write the query
3:08

again without a window function a lot of times when uh they're asking you these kind of questions as a data engineer

3:15

they're mostly just trying to see um and make you squirm a little bit and see like oh like how much depth does this

3:22

guy actually have can he solve the question in a completely different way can you give me can he can he say tomato

3:27

and tomato because if you can't say both he's obviously not qualified for this job and it's kind of absurd it's kind of

3:34

absurd I found that that that part of the SQL interview to be kind of absurd um another one that I've been in i' I've

3:40

I've had to do a couple times is uh using recursive CTE I don't know if

3:45

y'all have ever use like the with recursive uh command um I've literally done that in

3:52

interviews I think three times two two three times something like that and I've

3:58

done it um on the job zero times never not not even once it has not come up not

4:03

even one time so uh that's a great example of one that is literally like asked a ton as if it's like something

4:09

that you're going to need and really the only time that you would ever need that is in the case of where you have like a

4:15

parent child uh query around like managers and reports that's the only

4:20

time I've ever really seen it was when you have this crazy like tree hierarchy that's the only time you ever really need to use recursive CTE um people like

4:28

essentially think like oh if you know recursive CTE that's that's when you know right that's when you know that you

4:34

are the that you are the head honcho right you are the sequel wizard you have officially you are now level 100 at

4:41

squel and you can get you know you can get your cape and celebrate and it'll be awesome um I don't know I uh I'm

4:49

skeptical of it because of that um correlated sub careers are the same way like uh I I feel that there's uh like I
4:56
it's one of those that like I've never needed to use actually in the job where you can leverage joins and aggregations
5:03
and all that stuff just like keeping things simple keep things simple right and like that's one of the other things
5:09
about like the recursive CTE where it's like even if you wanted to do recursive CTE for a management a manager and
5:17
Report structure the thing is is most companies they only have a couple layers right so instead you could just do two
5:24
or three joins where you just join each layer of the company and uh you just do it that way and instead of using a
5:30
recursive CTE and that would solve the problem probably more efficiently and uh
5:35
but my whole point here is they're going to ask you a lot of stupid
5:41
stuff like a lot of things in the sequel interview that are going to just not be
5:46
relevant uh anywhere else and these things fall under the guise of advanced
5:53
SQL and because they're like well if you've never used recursive CTE before you're not a senior data engineer and
5:59
it's like bro I've never used that in my entire career and I'm a Staff dat engineer I've been doing this for 10 years so like I don't know man like I
6:07
think that some of these some of the questions that people ask need to be better like they need to like improve and they need to uh not necessarily uh
6:15
make those hard assumptions that like you have to have like one specific nugget of knowledge I have noticed
6:21
though especially between uh when I interviewed at Facebook and when I interviewed at Airbnb cuz there was like
6:27
5 years between those two interviews and um I noticed that like a lot of the
6:33

at least in big Tech that a lot of the the the sequel questions have gotten more like more relevant and but still
6:41
tricky more relevant and but still like the same level of tricky but not obnoxiously Technical and like
6:47
obnoxiously doing stuff like this cuz I I remember like uh when I was interviewing at Airbnb and I and I got I
6:54
got my sequel question it was like I still I solved it with a self join it was definitely a self join that I solved
6:59
with and I I probably could have used a window function but like I lean into not
7:05
using window functions at all when I'm doing uh SQL interviews unless there's
7:10
like specific keywords like if they say rank or they say rolling or they say
7:15
like very specific words in the question then those words are like they tip me off to say use a window function because
7:22
everything else is going to not be worth like trying to trying to do it with like antsy sequels not going to be worth it
7:27
and you're not going to get through the problem in time and and they're not going to ask you to do that either because it's too complicated in like a
7:33
30 or 45 minute setting so anyways they're different and one of the
7:40
things that I hope if any of y'all have failed uh um a sequel interview before
7:45
that that they that that is not that doesn't necessarily mean that you are not Advanced enough at SQL I've failed
7:52
squl interviews before like like I had some crazy ones some crazy ones with like where they asked me just really
7:58
ridiculous stuff and then I'm was like dude that was okay it was not meant to be that company obviously only has
8:04
massively hierarchical data and that's the only data sets that they have and that's why they ask every data engineer that question even though that's
8:10
definitely not true but anyways let's uh let's go to the next slide okay there are some things about
8:17

the de interview the de SQL interview that they do get right uh a lot of times you're going to get pressed in the de

8:23

interview around uh uh table scans like how many times are you scanning the

8:28

table because that really does make a difference for the um uh like the

8:33

optimization and the performance of the query number number of table scans is the number one uh uh like thing for the

8:41

performance of a SQL query and so that's going to be a big thing to think about um one of the one of the slam dunk

8:49

things that you need to do in your data engineering SQL interviews and elsewhere

8:56

like you're going to you're going to use these a lot in a lot of places in life as you doing count case when so you put

9:02

a uh some sort of condition in your aggregation statement and that is going to be uh just such a good thing I've

9:08

used that so many times to Pivot out data to do all sorts of really awesome SQL queries that are like fast and

9:15

efficient and scan the table once and uh that you can do really really powerful things with those so definitely if you

9:22

haven't tried out count case when I'm pretty sure you have cuz like y'all are pretty experienced but if you haven't

9:28

definitely do that and obviously cumulative table design comes up again where uh where in this case like you can

9:34

have queries that are cumulative and then uh you you show them how you would

9:39

then incrementally build it up and incrementally solve your problems that way so that's another thing that I've

9:46

done a lot like with uh um in the interview process is talking about how to query the minimum amount of data to

9:54

solve your problem uh then obviously there's like writing clean code clean

9:59

code is just so important like don't like especially as data Engineers like

10:05
dat that's our whole job like other people can write terrible SQL monstrosities but that's like we should
10:11
take pride in writing good SQL and obviously commentable expressions are your friend it's like the with keyword
10:18
uh use that a lot uh pretty much always especially in the interview if you use with a lot you're going to be good there
10:24
are some edge cases with uh comment table expressions with like postgress and older versions of postgress and
10:31
MySQL and stuff like that where like it can hurt performance but like in the
10:36
interview always use it because it's always the cleanest way to write the code and then use aliases don't ever have a column name that's like not Alias
10:43
especially if it's a derived column because like uh then your result set is going to look ugly and obviously you
10:50
don't want an ugly result set so uh those are the things that like I've noticed that de interviews kind of nudge
10:56
me towards they really nudge me towards actually doing these things and actually caring about efficiency so that's where
11:03
it's like I like them but I don't like them because like they're kind of a wild card where you can be asked like some
11:09
crazy question as well so anyways that's kind of the idea behind de interviews
11:14
and how they will um uh kind of nudge you in the
11:20
right direction for uh writing a good SQL which some of this stuff I wouldn't
11:26
consider Advanced but it's more of like thinking through your queries and I
11:31
think another big thing that you can use here uh not necessarily in the interview
11:36
but like it will help you get better at writing queries is this this thing called the explain keyword explain is
11:44

very powerful it shows you uh the query plan so it shows you how the query is going to be ran underneath the hood and
11:51
if you can uh learn more about the explain keyword and then like use it when you're writing your queries and
11:57
then understand like oh this query query creates this plan and this query creates this plan and then kind of building up a
12:04
fundamental understanding of how queries like how SQL is turned into this it's a thing called an a is an abstract syntax
12:12
tree which is essentially one layer lower where your SQL gets converted into
12:18
an a which is going to be all of the different uh
12:23
like filter aggregation join sort of like mechanisms that are within SQL and
12:30
The Ordering of them because the tree has to point right where you have like because you know there's the order in
12:35
which the SQL keywords operate in right where from and join happen first and
12:41
then where and then Group by and then having and then uh and then select and
12:46
then order by or whatever there's like that kind of ordering that a SQL happens in and so that is another part about the
12:53
abstract syntax tree that if you understand how the ests are ated with
12:59
SQL you're going to be just a way better uh practitioner of of SQL cuz I've
13:07
noticed that like once I started like digging deeper into that kind of stuff I like the odds that I wrote a a bad
13:14
performing query just like went away for the most part and I was like wow okay like I I know what's going on here even
13:21
complicated queries so that's an important piece of the puzzle as well so we're going to talk more about
13:29
about this stuff and we're going to use some of this stuff in uh the lab today as well um so there is a a suite of
13:39

three um Group by uh mechanisms here there's one called grouping sets one's
13:45
called Group by Cube and one's called Group by rollup and we're going to talk
13:51
more in depth about these it's a way to it's these are essentially a way to do
13:56
multiple aggregations in one query without having to do like nasty unions
14:03
so you can like group by like you can imagine like you can Group by like gender and Country and then you can
14:09
Group by just gender and just country and then overall as well if you want and you can essentially pick all of the
14:16
different rollups that you want uh and we we'll go into more details there um
14:21
self-join is another important concept that I think is often times uh brushed
14:28
aside when um thinking about SQL and like because most of the time when
14:34
people think about uh data modeling they think of like Kimble which has like facts and dimensions and joins but it
14:41
never really talks about how like a table can join with itself and so this
14:48
uh use case here a lot of times like what we're going to work on today at
14:53
least with in terms of self joins is is great as one of the things is like Sundar doesn't even realize how how
14:59
grateful I am for what he was talking about so he talked about funnel analytics today in the in his Boot Camp
15:05
or in his uh presentation and we're going to be creating a funnel we're going to look at funnel and conversion
15:11
rates on my website using self join so it'll be that'll be a fun uh little
15:16
project we're going to work on and that's where a table is joined to itself so that will be fun um obviously window
15:23
functions window functions are very very critical uh in the data engineering
15:29

interview I want to talk a little bit more about like the signals uh that you want to look for when when you should
15:37
know to use um a window function so you
15:43
have different things like if if they say monthly average they almost always
15:48
mean rolling monthly average and if they mean rolling if it's rolling monthly
15:53
average then uh window function is going to be your best bet and like that's going to and you're have to like B bust
16:00
out like the whole window function you're going to need to do like um average over and then probably partition
16:06
on some sort of Dimension and then order by date and then it'll be rows uh between 30 proceeding and current row
16:14
and that will give you the um that the the lagging uh 30-day monthly average
16:20
and then obviously uh you also have a thing called a cross joint unest and
16:26
lateral view explode uh unest if yall remember from weeks one from week one
16:31
and week two yeah we did a little bit in week two as well unest is essentially how you can turn an array column back
16:39
into rows and it essentially explodes out the array into rows that's why it's
16:44
called explode sometimes it's called unnest or explode cross join unnest and lateral view explode are actually the
16:51
same they literally do exactly the same thing uh it just depends on the query
16:57
engine um I personally like cross join unest as the canonical word so it's like
17:04
essentially how it works is like cross join unnest is like if you're working in like a SQL focused engine like Presto or
17:11
postgress or MySQL or Oracle or like it's very SQL based you're going to
17:17
you're going to see cross joint unest and if it's jvm based like uh spark or
17:23

Hive or any of like those kind of more big data oriented map reduce any of those kind of oriented Technologies then
17:31
you're going to see lateral view explode that's what um because lateral view explode was birthed out of like Hadoop
17:37
and cross joint unest was more birth out of like post out of like the postgress kind of syntactical stuff so um we're
17:46
not going to cover those today but because we already did and yall have been doing Advanced SQL techniques
17:51
throughout this entire boot camp and uh I hope that the homeworks in weeks one and two are also helping you get even
17:58
more more up to dat and more fresh on some of these Concepts so let's go let's
18:04
let's dig a little bit deeper into these this first one this aggregation thing I was talking about well let's talk about
18:11
um how this works so this is uh the most complicated of the three you have
18:17
grouping sets so you can think of this as so in this um query we are going to
18:23
this is actually going to give us this is essentially doing four queries in one so we're going to group on OS type
18:30
device type browser type like all of them together and then OS type and browser type and then just OS type and
18:36
just browser type so this is going to actually do four aggregations the way that you would do this without grouping
18:42
sets is you would need to copy and paste the query four times and then you would say Group by OS type device type browser
18:49
type and then Group by OS type browser type and then Group by OS type and browser type and you're going to need to
18:54
put dummy stuff in there for the the values that are not in the group by like
19:00
so for example when you're grouping by just OS type then uh if you want the union all to fit with all the rest of
19:06

the query you're going to put like overall for device type and overall for
19:12
browser type and because uh because those are just ignored in that
19:17
aggregation so they are essentially just completely not even looked at and so one
19:23
of the things that's really nice about this way of doing things is you can um
19:29
one you get performance benefits because Union all is one of the slowest and most terrible keywords in all of SQL so you
19:37
get a nice uh performance bump from uh doing that also you get a nice
19:43
performance or you you you get a readability bump because and you won't like you don't have to copy and paste
19:48
the query five times so you you only have to edit the query in one spot which is a nice uh maintainability readability
19:56
bump as well and then you also get access to whatever uh combination of um
20:01
Dimensions that you want to look at so one of the things that's important when working with this pattern
20:10
is when uh a when a dimension is ignored
20:15
in one of the grouping sets like for example in that last one when we're just grouping on browser type that means that
20:21
device type and Os type are both ignored and what grouping sets does to those
20:27
columns in the query is it nullifies them so one of the things that's
20:32
important to do before uh using grouping sets is you want to make sure that these
20:39
columns are never null because they get turn they get nullified uh when they get
20:45
excluded from one of the group buys and then the problem is is then you don't know if it's nullified because it's it's
20:51
excluded from the group by or if it's nullified because it's it came in null
20:57
so uh uh kind of one of the best practices here is before you use grouping sets or any of these grouping

21:04
patterns for that matter you want to coales all of the um the grouping uh the
21:12
grouping Dimensions you want to coales them to things like unknown or like some other value like unknown na or whatever
21:19
you want to call it and uh because then you'll know you'll know in that those
21:25
cases unknown essentially means that it was null and then uh if it's null that
21:31
means that it was excluded from the group by on one of the the the fewer Dimension
21:38
groupings if that doesn't make sense don't worry we're covering this in depth in the lab today so this is um so
21:45
keeping in mind this is the most complicated way um that you can use
21:50
grouping sets it gives you the most control because you you pick all of the
21:56
all of the grains that you want to aggregate along um let's let's take a little bit of a
22:01
shift here and talk a little bit more about Cube because cube is another interesting one so so you'll see cube is
22:09
a little bit um simpler um Cube essentially what Cube does is it gives
22:15
you all possible permutations here so in that case uh one of the ways that you
22:22
will know what the the number of permutations is it's actually a lot so
22:28
um uh so how it works is if you have three what it is is you get um you're going to
22:35
have a like there's going to be the combination of three and then you have the combination of two combination of one and then none right and so that's
22:42
going to give you all the possible uh values that you could have right so uh
22:48
like in that case you're gon to have all of them and then you're gonna have you got to pick two and when you pick two
22:54
right there's going to be three possible uh values right when you pick two and
22:59
then when you pick one there's going to be three possible values again because it's just any one of them and so that's

23:06
uh 1 + 3 + 3 that's seven and then you also need to pick zero so that's when
23:12
you just do the overall Aggregate and uh that will be where you essentially Group by nothing and so in that case this one
23:20
is actually this it looks like this is like one this one line is actually doing
23:26
eight different grains of data and so it's very powerful because like
23:32
you see compared to the grouping sets line right where you have to specify each of the grains uh it with Cube it
23:39
just gives you all of them automatically but this can also lead to problems especially if uh you one you pick
23:48
more because uh you saw how like if you have three dimensions you get eight
23:53
right but if you have four dimensions it it it blows up it blows up and like you get so like cuz it the number it's a
24:02
combinatorial explosion right so you can only really use Cube on like I generally
24:08
my my perspective is three like uh you should only use three dimensions uh
24:14
maybe four if you want to like get real fancy but that's going to you're going to have a lot of like different
24:19
combinations and permutations of things when you use four so I um I don't know
24:25
if I would recommend using four uh Dimensions here cuz the combinations just blows up definitely not five
24:31
because if you think about like five factorial right five factorial is 120 right and that's that's a lot like I
24:38
wouldn't want 120 grains of data so anyways that's Cube so there was one
24:44
more here that I think is important to talk about which is rollup so rollup is a little bit different in the fact that
24:52
how it works is that you use rollup for hierarchical
24:57
data um where imagine if you have like country and then you have state and you
25:03

have City so in that case we only like we always want to group on country and
25:09
then sometimes we'll group on uh country just country or then we'll group on country and state and then we'll group
25:16
on country and state and city so in this case like for this rollup it would be we
25:22
group on OS type by itself we also group on OS type and device type and then we
25:28
also group on OS type device type and browser type so rollup is a little bit different you don't get all like so cube
25:36
is like cube is a lot like generally speaking uh from my perspective when
25:41
using these three different ways of doing Group by rollup is going to be uh
25:46
like probably a better choice because with rollup right you just get the number of Dimensions is also equal to
25:53
the number of aggregations that you get so it grows linearly it doesn't grow um like factorially or com I guess
26:01
combinatorially so in that way cube is just not very scalable but it is a very expressive way to uh like talk about a
26:10
bunch of uh possible permutations of Dimensions um so like but generally my
26:17
experience I don't like I have never used cube in uh Big Data like in uh like
26:23
at Facebook Netflix Airbnb I've never used Cube uh I've used grouping sets at all the companies and I've also used
26:30
rollup at all the companies I've used both of those but I've never used Cube and the main reason for that is because
26:36
of this like crazy like it's like because one of the things about Cube
26:41
that I don't like is that it's going to do all the combinations even when
26:47
there's some that you don't even care about and that's the part about Cube that I don't like is it gives up it it's
26:54
it just does everything it in some combinations you probably don't care about and if you don't care about some

26:59
of the combinations then like you're wasting compute whereas rollup is nice cuz it's very clear that it's only going
27:05
to do three and then grouping sets gives you that very fine grain control over like I want this grain this grain and
27:12
this grain and for me as an engineer I am more of the I I don't like uh what's
27:18
called autom magical stuff like where like you write one line of code and then it magically produces all sorts of crazy
27:25
stuff cuz I think that um automagical stuff is one hard to debug and two like
27:32
um it's going to you lose control you like in this case like cube is not going
27:38
to be as efficient as here if I only care about these four Cuts then why
27:44
would I use Cube and get eight right why would I use Cube and get eight if I only care about these four so that's one of
27:51
the things that you want to be careful about is uh how um how much control that
27:57
you are giving up by using this expressiveness um so in this case my perspective is you want to use grouping
28:05
sets or roll up and kind of just ignore Cube unless it's like small data and you want to like like I think cube is like
28:11
the one place that I would I would say cube is like okay is when you are doing like exploratory data analysis and
28:18
you're just trying to like look at all trying to really understand all of the dimensions really quickly CU that will
28:25
give you a lot clearer picture into like what's going on and then you don't have to like use grouping sets because like
28:32
that's annoying because imagine having to put all eight in here for grouping sets just so you can understand what's
28:37
going on um so those are the three those are the three kind of like specialized Group by
28:45
keywords that like I have found to be very useful uh there is one caveat I

28:50
want to talk about here is if you're using big query you only get rollup that's it you don't get like big
28:58
query does not give you grouping sets big query does not give you Cube it only gives you rollup so uh just letting
29:04
y'all know uh if you're like big query fans okay okay so I want to talk a
29:11
little bit more about window functions window functions are all are just just an very important part of analytics uh
29:18
we're going to go super deep into window functions today in the lab uh one of the things about window functions is there's
29:25
just a couple pieces to them uh you have first you have the function us just
29:30
going to be like rank sum average dense rank row number um lag
29:37
lead I think first value last value I think that's it it's pretty much
29:43
it there's not that many there's like 10 and um so like one of the things I want
29:49
to talk about real quick is like there's three that are important to talk about where you have rank dense Rank and row number uh which essentially do the same
29:56
thing they kind of like they give an ordinal they give an ordinal output based on another column and like my
30:04
perspective is never use rank ever because like rank just like skips values
30:10
and stuff like that and like it's super obnoxious like I hate that rank skips values so like uh and there's no benefit
30:17
for that I mean unless like you're okay the one exception is like if you're in the Olympics and then there's like a tie
30:24
for second place and then uh then the next place is not third it's Fourth uh
30:29
because of that uh I think that that's an okay spot like but that's literally if you're a data engineer on the
30:35
Olympics like there is like I guess an extreme Edge case where rank is better but generally speaking I use dense rank
30:42

or row number every single time I would say I lean into row number more um
30:47
because it gives a unique value for each row even if there is a tie so if there's a tie for second uh so you have those
30:54
two rows that are both ranked as second for in row number one of them will be get will be ranked third and uh it's
31:01
based on the natural ordering of the data whichever row uh in the in the database is for is comes before will be
31:08
the one that's ranked higher um and so that is uh and then dense rank is great
31:14
because uh you do have ties but then you don't skip any ordinals because then it's like if you have a tie for second
31:20
then third place will be third instead of fourth because in rank it's Fourth
31:26
but in dense rank if there's a tie for second then third place is third and I like that cuz then you have like a
31:32
continuous numbering you don't have like weird gaps where like sometimes like third place doesn't show up and stuff
31:38
like that it's terrible terrible so anyways function big part of it uh then
31:44
you have uh the window because then you have like the over keyword so you have like function over and then you have uh
31:50
the window itself which has three things right you have the Partition by order by and rows Clauses so Partition by is
31:57
essentially how you cut up your window so like maybe you only want to look at Windows per user or per country or per
32:04
date or per month or whatever there's like all different ways that you can cut up the window and that will help you
32:11
define things uh then you have order bu which is how you sort the window uh this is great for ranking functions uh and
32:18
also for uh you need to have the order by in there if you're doing like rolling sums if you're doing like a a like a a
32:25

monthly rolling sum or something like that then you're going to need to have a good order by CU otherwise it will just

32:32

pick the last 30 rows and those that might not be the right 30 rows so you definitely need to think about order by

32:38

when you're doing rolling uh functions and sums and averages as well and then

32:43

you have the row clause which essentially just determines how big the window can be uh you can you can have

32:50

the window be the entire window and that's like what you can say is and we're going to look at this in the lab

32:55

but there's like so you can have rows between and then it can be unbounded proceeding and unbounded um uh following

33:04

and that's just in that case it's give me all the rows before and after the current row and so then you can get the

33:11

entire window that way and you can understand like how to do like a sum of the entire window and so that can be

33:16

another really powerful way so the default value for rows right is between

33:22

unbounded proceeding so unbounded proceeding is the default value for um

33:28

uh the proceeding like the the left side of the window and then the right side of the window is current row so it's always

33:35

like backward looking it always just looks from the current row backwards and it will go from the current row backwards in time so if you do like a

33:42

sum with a partition by and order by without the rows Clause you're going to essentially get a cumulative sum up to

33:49

that point in in the rows so like say you have 30 rows of data and you're looking at the row that is the 30th row

33:56

and you're summing then you don't use the rows Clause you're going to have just all the data in that window from

34:02

that row backwards and so like if you don't Define the rows uh um Clause then

34:09

your default is you get accumulation where you just get um all the rows before the current row based on order by
34:17
this is the idea for window functions um I'm sure some of y'all already know this uh we're going to cover window functions
34:22
a little bit in the lab as well today so um uh we're going to look at data modeling versus Advanced SQL so um
34:31
obviously like if you are a data engineer and you're just given raw data and it's and then you says like find
34:36
Insight in this data like you can probably do it right you might have to write a lot of SQL you might have to
34:41
create a lot of temp tables you might have to run some pipelines so um this uh image here uh that I uh I posted this
34:49
like funky image so what it was was uh I went to do e and I uh used I punched in
34:57
SQL gymnastics and uh that's what I got like that Dolly said that that is what SQL
35:03
gymnastics is and I'm like okay thank you AI thanks for making this presentation better so what can happen
35:09
here is uh if you are working with a data analyst and they are working on
35:15
your tables one of the things is is like you as a data engineer a lot of the time
35:22
no SQL very well or you should and you will after this boot camp if you don't and um one of the things about that is
35:30
you sometimes might have the expectation from your uh analytics partners that they have the same Proficiency in Sequel
35:35
so that they can just tackle whatever data model you give them and that's a lot of times not the case a lot of times
35:42
like uh these people are going to be uh uh they don't know all the crazy new udfs or functions or stuff that make it
35:49
easier to Crunch this data and you need to give them uh the ability to just like
35:54

understand it easily and if you if they need a like have to like for example if you give them a table that's not duped
36:01
and then it's like okay just use row number to DD it right you could say that as a data engineer and just like Advocate yourself from all
36:06
responsibility right and just be like no like all quality eras are on you when you're querying and so and then you
36:12
could just use your row row your row number function and uh window functions and I'm just going to give you garbage
36:19
and then but I I it's on you to fix and uh that can happen I mean obviously like
36:24
if you do some amount of data modeling and it doesn't even have to be that much and data modeling and data quality like
36:29
what we covered last week then uh a lot of these problems will disappear and you'll be able to have data that like
36:36
makes it so analysts don't need to do Advanced SQL but obviously sometimes analysts like this is not necessarily
36:44
always the case like sometimes analysts want to do Advanced SQL because they're trying to like they're trying to answer
36:49
a very very specific question that like you aren't going to write a very specific pipeline for because that's
36:54
like a waste of time and so like I'm not saying that like just because a data analyst is doing crazy gymnastic SQL
37:01
that means that like you did bad data modeling but it could be it could be it can be a signal so that's one of the
37:08
things where like if analysts are struggling to like get insight out of your data that's something that you want
37:14
to take in and be like oh maybe I need to uh revamp this table or I need to change how it is or I need to do
37:20
something to it that will make it so that the analysts can understand things better this becomes even a bigger
37:25

problem as as the data scale because analysts will want to do more complex queries on longer time frames

37:33

and when you have uh and if you have more complex queries on longer time frames uh of big of bigger data then uh

37:41

you that's just a lot of complexity each one of those is complex by itself so when you stack complexity on top of

37:48

complexity on top of complexity you're going to get slow queries queries that don't run queries that fall apart

37:55

queries that take forever right and uh you don't want that you want your analytics Partners to be happy right and

38:01

that's where're working with them on like if they want to try to solve those complex questions and they're not like

38:08

one-offs that's where it's important to understand the analytical patterns that your your analysts and your analytical

38:14

partners are trying to do so that you can provide data sets that are

38:20

delightful to use in that way similar to you know in week two when we were talking about like the long-term

38:25

analysis stuff that I did with like the the fact Dimensions like or the fact arrays where you have like that long

38:32

array metric that was that's a great example of that where I noticed that I was like hey my analysts are freaking

38:38

they're querying very long time frames of data and obviously I can give them

38:44

just the regular fact data or the regular daily data but like that's just going to still be really slow it's going

38:49

to be really terrible and so that's where uh you can you can think of this as like both on the terms of volume but

38:55

also on complexity because a lot of times they might do the same query over and over and over again and really what

39:02

you should do is materialize that and then they should just query off of that directly and then that would massively

39:07

speed up their analytics so a lot of times like one of the things I always do as a data engineer is uh and this is

39:15

something I do especially when I first start on a job is understand where the analytics partners are at and like what

39:21

they are querying and what they're building and what they're presenting just so I can understand like where some

39:26

bottleneck are and like how I can potentially uh speed up their processes because if you can make analysts faster

39:33

as a data engineer you're doing your job you're doing a great job at doing your job and that's uh like you want to do

39:40

that okay so uh this is I think we're almost done here and we can take a

39:45

transition to the lab here in just a second so you have a symptoms of bad data modeling right you have slow

39:50

dashboards uh this is where grouping sets can be really useful because if you are using like row or daily level data

39:58

um like um in your dashboards and it's not it's not pre-aggregated then your your dashboard

40:05

will if your company scales to a big enough scale then uh your dashboard is not going to work it's going to

40:11

eventually it's it's destined to not run anymore and so uh like if you use

40:17

pre-aggregation though and you and you only have your dashboard be um pre-aggregated data then your dashboard

40:23

will be infinitely scalable I noticed that like when I was working working at Facebook cuz it's like I was working on

40:29

uh this dashboard that like consolidated all the metrics between WhatsApp Instagram Facebook Messenger all the

40:34

different like apps and uh each one of those apps had like a billion users so it was like am I going to really load in

40:42
uh you know five billion rows into Tableau am I gonna do that like is
40:47
Tableau gonna is like is Tableau gonna do that or even even postest is postest going to do that is postgress going to
40:53
freaking uh um handle that and it's like obviously not and so uh what I did was I
40:58
figured out like kind of a a framework where you always pre-aggregate and then but you pre-aggregate along the
41:05
dimensions that people care about like whether that whether that be age or country or device or um app like we also
41:14
wanted to see like yeah we have this many WhatsApp users this many Facebook users so you want to aggregate on those
41:20
Dimensions because then that data goes from being billions of Records like and then with app it's like to like five
41:27
right so that's like going from billions of rows to five rows is like so
41:33
efficient you know so that's a thing to think about when uh like when you're
41:39
dealing with uh some of this like analytical data modeling is slow dashboards is like a very common symptom
41:45
that you need to pre-aggregate your data obviously uh another one here is going to be around like if you if your
41:52
analyst is like you look at their query and there's like 10 CTE in the query it's like okay like maybe there's a
41:58
staging step maybe there's another step in the middle here especially if you see these again and again and again and you
42:04
like they keep sharing these queries with you that are like freaking 10 CTE that usually means that like the first
42:10
five should be materialized as a temp table or as a staging table and then they they do their analysis on top of
42:16
that and maybe that table has short retention so that you don't like burn down the warehouse but like that's like

42:22
definitely do that don't like have your analysts just like suffer and always because remember this this is something

42:28
that I always think is very important to talk about with data modeling is storage is cheaper than

42:35
compute Right storage is cheaper than compute across the board cheaper so in

42:43
those cases where it's like if you have an analyst who's running 10 CTE if you materialize halfway through then and and

42:50
that's the table that they run off of then you pay a little bit of storage to save a lot of compute and then um even

42:56
if it was one: one right you pay the same amount of storage to get the same amount of compute that's still better because then the amount of time that the

43:03
analyst gets back is also better right because that's the other thing you got to remember is that like there's Cloud cost there's employee cost right of like

43:10
waiting for queries to run and then there's also storage cost right compute and storage cost and employee cost so

43:16
even if like if you're making the storage and compute tradeoff and the storage cost is the same as the compute

43:23
savings it's usually still worth it because you get a different benefit and

43:28
that benefit is going to be on the uh employee Time Savings and time reduction

43:34
and you get more a business velocity so that can be another massive thing another big uh thing right is case when

43:40
statements like that just means that like if you have an analyst is doing all sorts of gnarly casewind statements that

43:45
usually means that your data model is not robust enough or it doesn't it's not conformed enough you aren't conforming

43:52
the values to what they need to be so those are the the big ones there's going to be a lot of other ones that you could

43:58
probably think of uh but those are going to be the three big ones that I was thinking about when I was thinking about

44:04
like how I work with analytical Partners so yeah like uh remember that that like
44:10
bad data modeling and like I wouldn't I wouldn't I don't know if I'd call this Advanced SQL I'd probably call it more
44:15
complex SQL which is probably a better word to use on I'll probably change the slide here before I upload it but uh
44:22
that those things kind of are connected and that like it's your job as a data engineer to make it so that your analyst
44:27
has simpler queries especially if they're running those queries a lot congrats on getting to the end of the
44:33
day two lecture you're taking this class for credit make sure to switch over to the other tab so you can get credit