TRANSCRIPT · EDB POSTGRES AI SUMMIT SEOUL 2026

Why We Bet on “Agentic Lakehouse”?

Jeonghwan Lee, CEO, Shopcast

EDB Postgres AI Summit Seoul 2026 · September 3, 2026 · Sofitel Ambassador Seoul

Hello everyone. I’m Jeonghwan Lee, CEO of Shopcast. Being invited here today is quite an honor. But I thought hard about it. Why, when Samsung Electronics and IBK Industrial Bank of Korea — all these really big companies — are here presenting, did they invite me? So I asked, and here’s what they said. The first time we — I — came to know EDB was June of this year. In June I encountered EDB, was introduced to a product called EDB WarehousePG, and then, through Dataworks, installed it on our company’s servers.

And applying WarehousePG to our actual service took exactly three days — we completed everything, including the full migration. So EDB told me that of everyone who installed the product, we were the fastest case — installing and migrating without asking a single question. “Ah, so that’s why I got invited.” So I figured it would be best to build my talk around that story, and that’s what I’ll focus on. First off, I’m not a DBA, and I’m not a developer either. But — maybe that’s exactly why I was invited today. For the actual deployment, I did everything entirely by myself, without any help from our company’s DBAs or developers. So let me unpack that story for you now.

Here’s the order I’ll go in. Since some of you may be hearing of Shopcast for the first time, I’ll briefly introduce our company, then what happened, and how we applied this product — as a real-world case. First of all, here’s some data. When it first landed on us: 7 billion records. One of the main services our company runs accumulates about 7 billion log records a year, and about 1.8 billion per quarter. Normally, when we run royalty settlement and distribution for the associations, it’s done quarterly — and with WarehousePG we pulled off the historic feat of finishing that in just seven minutes. I wanted to show that off and share it with you. First, let me introduce our company. Our flagship service — the company is called Shopcast, after all — is providing music to stores. Individuals use Melon or YouTube for their music, but businesses and stores typically use a service like Shopcast for their in-store music. Our second business is large-format LED displays — if you ever need one, please reach out. And for broadcast monitoring: everything airing in Korea — TV, cable TV, radio — every broadcast going out, we record and capture, detecting in real time which songs aired and how many times, and monitoring all of it. And naturally, copyright-related music work — royalty settlement and distribution for the rights associations — we’ve done for decades. We’ve also handled settlement for music streaming sites — Korea’s big, major music services. And distribution settlement — not general retail, but settlement for music distribution platforms. We make music too, and we hold copyright technology, doing content-based and recognition-based identification services — so we’re a music-and-copyright-technology company. And since that means handling lots of logs, we packaged ourselves up a bit for this talk: “a company that turns playback logs into money” — anyway, that’s the kind of company we are.

Running these services, we serve about 70,000 stores nationwide. One store produces about 300 log records a day. Convenience stores actually run 24 hours, so they produce even more, but on average: 70,000 stores, about 300 a day — I did the math, and... about 21 million — 21 million records a day. Add that up over a year — over a year, you get 7 billion log records accumulating. And in the past, for settlement and distribution, all of this had to be reported. Now, they say if you pre-aggregate as you accumulate, it's not a big problem — but we have to submit real-time logs to the Copyright Commission every single day. We submit real-time logs.

And now — earlier I mentioned our music-site work — on the music settlement side, about 1.8 billion records arrive every single day. "How do we deal with this? This is too hard" — that's where we were when we sat down to think. When we first faced this, the very first thing we tried — back in the day, the question was how to settle and distribute about 20 million records, and we ran settlement on an enterprise Sun machine taller than me. But one settlement run took 21 hours. Got it wrong? Another 21 hours. That's how we worked. What the developers did was split the data. Split it, GROUP BY, gather the results, GROUP BY again, and again — settling that way, until we thought: "what are we even doing?" And that's when we met something new. We met Hadoop. Wow.

It was a new concept for us — a whole new world opened up. We tried Hadoop, and — as you'll know — some CEOs here will relate — "hey, there's this thing called Hadoop, apparently it can do settlement — let's try it." But nobody at our company would do it. So I chased it down myself — went to another company and took about a week of training, studied it myself, and learned how to use Hadoop and Hive.

I showed our developers: run a GROUP BY like this, and the data comes out — and they were stunned. Jobs that ran 21 hours, 16 hours, started returning a GROUP BY result in about thirty or forty minutes. "Wow — this actually works." We ran on that setup until this June, but it wasn't without problems either.

You need someone who can operate it, and with Hadoop there's HDFS, right? Yes — you have to understand that foundation, and there's a lot you need to know. And to make it work alongside what we're used to — Oracle, Postgres — there's a lot of extra plumbing, and so employees kept coming with questions to me — who barely understood it myself — and difficulties kept piling up. And then, this June, we came across WarehousePG. Ah — apologies — let me stay on the Hadoop and Hive pain a bit longer. Using Hadoop, here's another problem we hit: it holds your data in three copies. "No — can't you hold just two?" No — it absolutely must be three. Honestly, I don't know why. It just has to be three, they said. Fine. But those three copies — I want to type ls -al or dir and have the data just be there — and it isn't. The data supposedly lives in some virtual space. Unsettling, right? So to keep it safe, we also copied it to external drives, and onto big storage as well. In the end, to preserve one set of log data, we kept three copies plus more — five sets in total. And the data we pile up in a year comes to about 30 terabytes.

Trying to keep five copies of that was just overwhelming. So — to protect one set of logs, we had five copies, and add RAID configuration — you know how it is, RAID arrays just dying — yes. Altogether about six-fold consumption and waste. And then there was another problem: the jobs that used to

take 16–20 hours did come down to about 40 minutes of processing with Hive. Processing was fine — but to actually learn anything?

You ask a developer, “please pull this” — “sure” — and then it takes forever. Only that developer can do it. Now me — I’m a little different: GROUP BY, SELECT, I can write those a bit — HAVING and the like I use fairly well — so I’m the type to dig in myself, but even then nothing was visible at a glance — it had its painful sides. That was our life until this June — though in this industry, it was a revolution.

“They settle music royalties with Hive and Hadoop? Insane!” — and nearly all the settlement work in the industry came flooding into our company. So when we met EDB, we told them three things. Don’t split the 1.8 billion records, don’t preprocess them — please, we just want to use them as-is. And operations and analytics should run on the same numbers. And — our company isn’t big. We can’t hire dedicated, high-end specialists, so give us an environment our existing people can use. That’s exactly what I asked EDB for — and the one thing that satisfied all of it at once was WarehousePG.

What I was thinking at the time was this: don’t touch how the logs accumulate today. Object storage — cheap storage like S3, or, well — inexpensive storage, you know — plug in hard disks, configure RAID, and there’s a petabyte — that kind of low-cost storage — keep our logs piling up there exactly as they are. Exactly as they are — you must not touch them. The moment you touch them, problems begin. Leave them as they accumulate, and on top, place the WarehousePG that EDB installed, and on top of that, use what we already know — SELECT, GROUP BY, HAVING — to see the data directly. That’s the system we needed. And this time — the reason I could finish everything in three days — let me share one thing: I used Claude Code. I set up a folder in Claude Code, and said: “Here’s what I need to do. We can’t use anything new yet — we’re still on Oracle. We’re on Oracle, and the final output ultimately has to land in Oracle. And the data coming to us looks like this. It has to be grouped and loaded into Oracle.” That’s all I explained. Then I contacted EDB. “I need to study — please send me the manual.” And they said — they were so sorry — there’s no Korean-language manual. That was a bit awkward, but fine. And then I received a 1,600-page manual. I took that 1,600-page manual and put it into the Claude Code folder I’d made. “Study this.” Really — “study this.” And: “my goal is to load the grouped data into this target schema in Oracle.”

We deployed it in three days. Nothing in the system had to change. Nothing at all — the existing system stayed as-is. We took what had been running on Hive and Hadoop, lifted the data out untouched, and loaded the final log output into our Oracle database, in exactly the form we wanted. Deploying all of that took three days. And in practice, the job that runs each quarter now finishes in seven minutes.

So — with AI, and with EDB’s 1,600-page English manual — that’s how it happened. And since I now use AI heavily, the most important thing I’ve learned: if you ask AI, “here’s the thing — we’ve got 50 billion records, and I need you to GROUP BY and build the data for settlement” — it hands you bizarre data. It does strange things — brings back impossible numbers. But we gave the AI a tremendous pair of wings: given the tool called WarehousePG, it uses this tool to build flawless data in a remarkably short time and deliver it. And — the AI didn’t have to keep running — it finished the

job. Every day, logs pile up in the exact designated folder, and it set up how those accumulated logs get processed, all the way to delivering into the desired target table in Oracle — it automated that entire pipeline, and then stepped away.

So — I came here to brag a little: this is how we deployed it. Yes. Let me move along a bit. For our in-store music, with EDB WarehousePG, it finished in seven minutes. Now — seven minutes versus forty: not much difference, you'd think? Forty minutes is "kick off the job when you leave for the day and come back in the morning"; seven minutes means if it's wrong, you just run it again. You can run it several times a day. Yes — that's the difference. And — an effect even bigger than the visible one: previously everything ran as batch jobs — they ran automatically, but the real problem is re-running when something goes wrong. And that became so smooth to handle, and our staff stopped working overnight — things like that happened, which I think is genuinely great. The logs that come from our clients, the logs we accumulate ourselves — we keep them all exactly as-is.

We keep everything as-is, and we never even need to touch it — the WarehousePG sitting in the middle takes the raw, untouched original logs and turns them into whatever output we need. Whether or not you ever touch the original data is genuinely, practically critical. And — here's an interesting case, so let me share it. One music service — see, when we GROUP BY the raw logs we've built up, they shrink to about 10 million rows. The raw is too heavy. Then, in the tools we're comfortable with — Oracle, MySQL, or Postgres — we can freely SELECT, ORDER BY, GROUP BY — my specialty, GROUP BY — and do everything we like. But here's the problem: when those 20 million rows meet the rights-holder data, they balloon back into the hundreds of millions. What do I mean? A song has its composers, its lyricists — and these days, a song doesn't have one composer. It has ten. And you know Produce 101 — "Pick Me Pick Me"? That song has 101 singers.

Yes — 101. So for one song, once the rights DB lists composers and lyricists — plus the tambourine player, the castanets player, the drummer — you end up with about 200 people. And the group that sang Pick Me later becomes Wanna One or — whatever it was — anyway, the remaining 90 members drop out and it becomes 11. So rights data that covered about 200 people later changes to just 11. Right — and then, with every one of those sequential rights changes folded in, matched per rights organization — the 20-million-row data balloons back into hundreds of millions. But our big, big constraint is this: browsing various statistics is one thing, but settlement absolutely requires exact data.

So — the common thread running through these cases is this: all those hundreds of millions of records — to Samsung Electronics or IBK, sure, a few billion rows may be nothing — cute little data, even — but for an SMB like us, billions of records is enormous data. An ordinary developer can't even open it. Earlier the results came out so fast I was honestly a bit jealous — for the much bigger companies here, this may look like fairly routine data, but for a company like ours it was genuinely difficult data — and through EDB WarehousePG, we were able to overcome it, like this.

And the thing we consider most important: we did not fit the data to the system. The data stayed put, and the system did all the changing — and through the middle of it all, Claude Code — my beloved Claude Code — was there. I can't really write SQL. GROUP BY — that I do well. Yes, GROUP BY I

do well. I can’t write SQL, but these days, our employees come to me asking me to pull data. Yes — “pull this for me” — and, well, that has now actually become possible. What I actually did — I told you earlier — I threw it at Claude Code: “From last quarter’s in-store playback logs in the lakehouse, give me playback share by region and genre, and find stores with anomalies versus last year.”

And it found them. It found them — and after that, I said “make me pretty charts too, build me an HTML page” — and it just did it all. Honestly, it built a page that looked wonderful to me. I also asked things like: analyze how IU’s songs are used in which stores, and what effect season and weather had on that.

It analyzes in remarkable detail. But what I used most was this: people around me actually make songs. When they release a song, most of them work hard to promote it. It used to be getting on broadcast; these days, they promote on social media, run ads on YouTube, and appear on shows too. And for the person who made the song, the thing they most want to know is: I spent money on marketing — what results did I actually get? So I tell them one thing: “Bring me everything you spent on marketing.” And they do. “I ran this YouTube ad on these dates, then this Meta campaign from this date to that, and appeared on this program” — they bring it all. Then I hand that data to the AI: compare it against the data in our lakehouse and pull the effect of each marketing channel for me.

And just like that, people come running to me — “pull it for me, pull it for me!” This used to be hard. One round of it used to take me a couple of hours — now I just say it — well, not say it, type it — and the data comes out, dramatically, and nicely — “and make me a pretty PDF while you’re at it” — deliver that, and they’re genuinely moved. Because now they know which marketing worked and which channel was the most efficient. For people who make, distribute, and market music, it’s been used incredibly preciously.

So — I thought about it. Why was this possible? We kept the originals intact — we never touched them. And — we accessed everything with standard SQL that anyone — even I — can use. And beyond standard SQL, it all ran through me lovingly talking to my beloved Claude Code. And then — operations and analytics both go through that one channel — that’s why it worked, I believe.

Ah — the Agentic Lakehouse. Our EDB WarehousePG. Yes — from a warehouse to a partner now. A partner. A partner that even makes proactive decisions. And what that workbench really is: the object storage we so comfortably piled data into, through WarehousePG, and then together with the many AIs I love. And — the AI agents work even at night, while we sleep. Around the clock, they monitor, detect anomalies, trace root causes, and self-heal. So — what has changed the most?

I used to go to developers: “build me this, pull me that.” If there are developers here — your bosses ask you for that constantly, right? “Pull me that Excel report.” Now: just give me a connection. Just hook it up — give me the access credentials and the database view definitions, wire it up to the warehouse — and instead of queries, just open the channel where I can hand things to Claude Code or an LLM. Do that, and those people will never come to the developers here with those requests again.

If I could go back three years and give myself one piece of advice: don’t throw away a single log. Just keep everything, exactly as it arrives. And choose standards — so your people don’t suffer. Later it all connects with AI, and these kinds of worries — in a case like ours — EDB and its partner companies will take care of it all for you, I believe. Now, with all of that, I believe we can do everything. Then what isn’t solved yet? Well — why AI, still — for settlement and distribution, the very last mile still needs a human hand. Yes, that’s a part worth thinking about. There’s much more I’d like to say, but I’m getting the signal from the back, so I’ll wrap up. In conclusion: our 7 billion records — about 1.8 billion per quarter — through EDB WarehousePG: solved in seven minutes. A dramatic story — and that concludes my talk. Thank you.

Edited transcript of the session, lightly adjusted for reading. Some passages have been omitted.