Engineer

Data warehouse

a database that gathers a business's data for analysis

What it is

A data warehouse is a database built to answer questions about a business rather than to run it. The systems that take orders, book shipments, issue invoices and log support calls each keep their own data, shaped for their own work. A warehouse receives copies from all of them, keeps those copies for years, and arranges them so that a question spanning several systems becomes one query instead of a week of exports. Building and running one is called data warehousing.

The difference from the databases it copies from is in the workload. An order system reads and writes a few rows at a time, thousands of times a minute, and keeps only the present: when a customer moves, the old address is overwritten. A warehouse is written in batches, read by queries that scan millions of rows at once, and keeps the past on purpose, so last year’s figures can still be broken down the way the business looked last year.

Ralph Kimball brought dimensional modelling to the warehouse industry in 1996, and the Kimball Group divides such a system into two rooms. The back room holds the ETL system, which they break into 34 subsystems grouped under four jobs: extracting data from the sources, cleaning and conforming it, delivering it to the presentation area, and managing the whole process. The front room holds the presentation area, which is what people actually query. It is organised by business process, kept at the most detailed level each process records, with aggregated summaries added for speed, and structured as fact and dimension tables. Around it sit the BI applications: ad hoc queries, standard reports, dashboards, scorecards and analytic applications. The Kimball Group calls the front room “the public face of the DW/BI system”.

That is not the only arrangement. The Kimball Group's own architecture page describes the Corporate Information Factory, a hub-and-spoke design that keeps atomic data in a normalised enterprise warehouse in third normal form and runs a second ETL step to load dimensional tables for reporting, and a hybrid in which an existing normalised warehouse supplies a dimensional presentation area. What every arrangement shares is the separation: the systems that run the business are not the place where it is analysed.

A warehouse is a role, not a product. It can be a PostgreSQL or Microsoft SQL Server database on a server the team runs, or a managed service such as Google BigQuery, which Google describes as a fully managed data platform combining a cloud data warehouse with analytic tools. BigQuery keeps storage and compute apart so each scales on its own, stores data by column rather than by row, and takes queries in ANSI-standard SQL.

Use the concept in a decision

Put "What is a data warehouse?" to work in a real decision. We can help relate the concept to the system you are building or buying. Sales, operations and finance figures brought together, so two departments stop bringing two answers to the same meeting.

Explore our services Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. Made a geographical data pipeline 50x faster — 50x faster geographical data pipelines

Made a geographical data pipeline 50x faster Get in touch →

Why it matters

A business that asks the same question of two systems gets two answers. Sales counts a customer when an order is placed, finance when an invoice is paid, and support whenever a ticket arrives. None of them is wrong for its own purpose, and none of them adds up with the others. The warehouse is where one definition is agreed and written down in the load, so that every report built on it counts the same way.

The Kimball Group's tool for this is the conformed dimension: one customer, one product, one calendar, maintained centrally in the ETL system and shared by every fact table that needs it. A sales fact table and a support fact table that both point at the same customer dimension can be compared line by line. Their bus architecture lays the business processes out against the dimensions they share in a matrix they call the architectural blueprint, and it lets the warehouse be built

one business process at a time, which they say shortens the time to market. A team can deliver orders this quarter and returns next quarter without redesigning what it already shipped, as long as both use the same dimensions.

History is the second reason. Operational systems keep the current state; a warehouse can keep every state. The Kimball technique for a changing attribute, called Type 2, adds a new dimension row with the new values and a new surrogate key, and every fact loaded from that moment on points to the new row. Sales from before a customer moved stay with the old region; sales after go to the new one. At least three extra columns, an effective date, an expiry date and a current-row flag, make it possible to ask what was true at any date.

The third reason is plain load. A report that scans three years of orders does not belong on the database that is taking today's orders, where it competes for the same memory, disk and locks as the checkout. Moving analysis to its own database lets each side be tuned for its own work.

The price is that a warehouse is a copy. It is as fresh as its last load and as correct as the rules in that load, and someone has to own both: the schedule that runs it, and the definitions of a sale, a customer and a month that the business agreed to.

What it looks like in practice

The heart of a dimensional warehouse is a fact table whose grain is declared first, here one row per order line, surrounded by the dimensions that describe it. The customer dimension carries the Type 2 columns, so a customer who moves gets a second row rather than an overwritten one.

```
CREATE TABLE dim_customer (
    customer_key bigint PRIMARY KEY,   -- surrogate key, owned by the warehouse
    customer_id  text NOT NULL,        -- the key in the source system
    region       text NOT NULL,
    valid_from   date NOT NULL,
    valid_to     date,                 -- empty while the row is current
    is_current   boolean NOT NULL
);

CREATE TABLE fact_order_line (
    date_key     integer NOT NULL REFERENCES dim_date,
    customer_key bigint  NOT NULL REFERENCES dim_customer,
    product_key  bigint  NOT NULL REFERENCES dim_product,
```

```
    quantity     integer NOT NULL,
    net_amount   numeric(12, 2) NOT NULL
);

SELECT d.year, c.region, sum(f.net_amount) AS revenue
FROM fact_order_line AS f
JOIN dim_date AS d USING (date_key)
JOIN dim_customer AS c USING (customer_key)
GROUP BY d.year, c.region
ORDER BY d.year, c.region;
```

Because each order line was loaded with the customer row that was current at the time, the query reports revenue by the region the customer lived in when they bought. Joining on `customer_id` and filtering on `is_current` instead restates all of history by today's regions. Both are legitimate questions, and the model answers either without a second copy of the data.

The promise of the data warehouse design service is exactly that first reason: sales, operations and finance figures brought together in one place, so two departments stop bringing two answers to the same meeting. At larger scale, the founder architected and ran 100 data warehouse databases on PostgreSQL, Microsoft SQL Server and Google BigQuery between 2022 and 2024, holding mostly GIS and time-series data such as spatial coordinates and sensor readings. The work combined normalised and denormalised table layouts with spatial indexing (PostGIS on PostgreSQL) and time-series partitioning on the first two engines, and time-partitioned, clustered tables with compression and columnar storage on BigQuery. Query response times for GIS and time-series data fell by 40 to 60%, and the platform let the organisation using those databases launch new data products such as real-time analytics dashboards.

Where it goes wrong

One fact table holds two grains, and every total counts some money twice. An order has lines, and it also has a delivery charge that belongs to the order as a whole. Put the charge on every line and a sum over lines counts it once per line. The Kimball Group's four-step design process puts declaring the grain second, straight after choosing the business process and before any dimension or fact is named, and their rule is that different grains must not share one fact table. Keep a separate fact table at order grain for charges that belong to the whole order, or divide the charge across the lines by a rule the business agrees to.

A customer moves, and last year's report changes. When the region on the customer row is simply overwritten, every past sale follows the customer to the new region, and a report that was signed off in January tells a different story in June. Treat attributes whose history matters with Type 2: close the current row and add a new one with a new surrogate key, in one database transaction so no load ever sees both rows current.

```
BEGIN;
UPDATE dim_customer
SET valid_to = current_date, is_current = false
WHERE customer_id = 'C-1042' AND is_current;
INSERT INTO dim_customer
    (customer_key, customer_id, region, valid_from, valid_to, is_current)
VALUES (90731, 'C-1042', 'North', current_date, NULL, true);
COMMIT;
```

Two teams build two customer dimensions, and the warehouse repeats the argument it was meant to end. Each new subject area is quicker to deliver with its own copy of the customer or product list, and each copy drifts: one drops closed accounts, the other merges duplicates differently. Build the shared dimensions once in the ETL system and make every new fact table use them. The bus matrix is the checklist: before a business process is added, it shows which existing dimensions the new fact table has to point at.

Reports are pointed at the staging tables. Raw copies of the sources land in the back room first, and once they are there it is tempting to query them directly because they are already loaded. Those tables carry each source's own codes, duplicates and gaps, and they change whenever a source does, so every report built on them inherits the problem the warehouse was meant to solve. Keep the back room for the ETL system alone and give reports, dashboards and analysts the presentation area only, where the conformed dimensions and the declared grain are what make the numbers agree.

Further reading

- Kimball Technical DW/BI System Architecture — the back room and the front room, the ETL subsystems, metadata, and the hub-and-spoke and hybrid alternatives.
- Enterprise Data Warehouse Bus Architecture — conformed dimensions, the bus matrix, and building one business process at a time.

- Four-Step Dimensional Design Process — business process, grain, dimensions and facts, in that order.
- Type 2: Add New Row — keeping the history of a changing attribute with a new row, a new surrogate key and three extra columns.
- BigQuery overview — Google's managed warehouse, with storage and compute kept apart and data stored by column.

Glossary

Choose for your actual constraints

Bring a representative dataset, the questions it must answer and the current bottleneck. We can agree how to measure quality, throughput and recovery.

See what it costs Get in touch →

Make the terminology practical

Ask how "What is a data warehouse?" applies to your organisation. A concrete example from your current system is enough to start the discussion.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/data-warehouse/