



The Synergy of Relational and Transformational Modeling

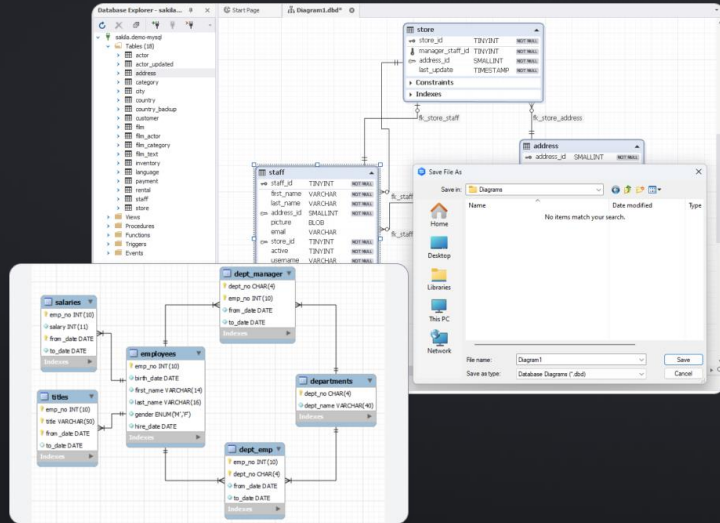
With some data mesh thrown in

Agenda

1. Why data modeling matters
2. Collaborating across the organizations
3. SqlDBM overview
4. Concurrent working and integrations
5. Relational vs transformational (SqlDBM vs dbt)
6. Data governance
7. Q&A

What is data modeling?

What people think it is



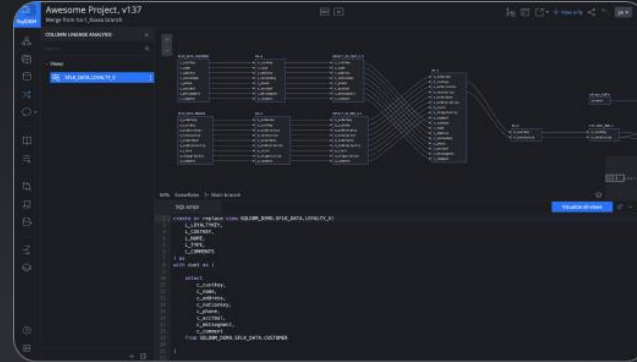
What it actually is



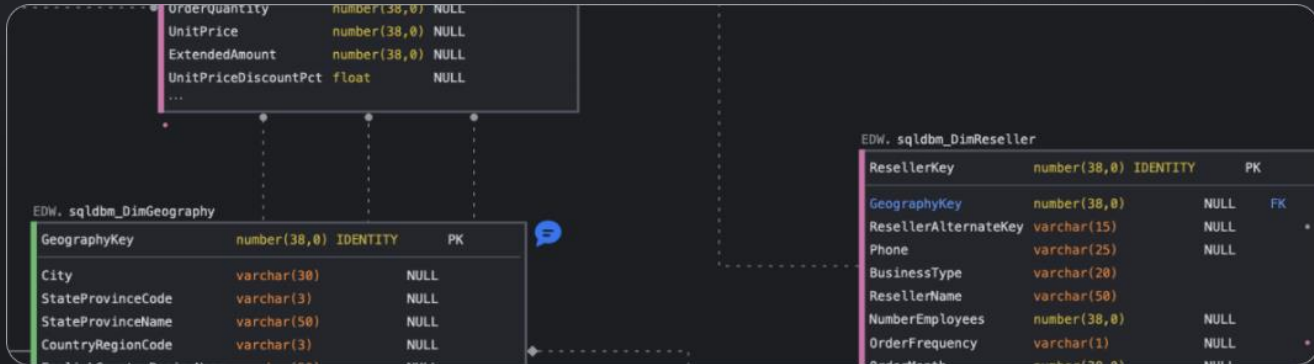
What else it is

```
SQL script
1 create or replace view SQLDBM_DEMO.SFLK_DATA.LOYALTY_V(
2     L_LOYALTYKEY,
3     L_CUSTKEY,
4     L_NAME,
5     L_TYPE,
6     L_COMMENTS
7 ) as
8 with cust as (
9
10     select
11         c_custkey,
12         c_name,
13         c_address,
14         c_nationkey,
15         c_phone,
16         c_acctbal,
17         c_mktsegment,
18         c_comment
```

What it also is

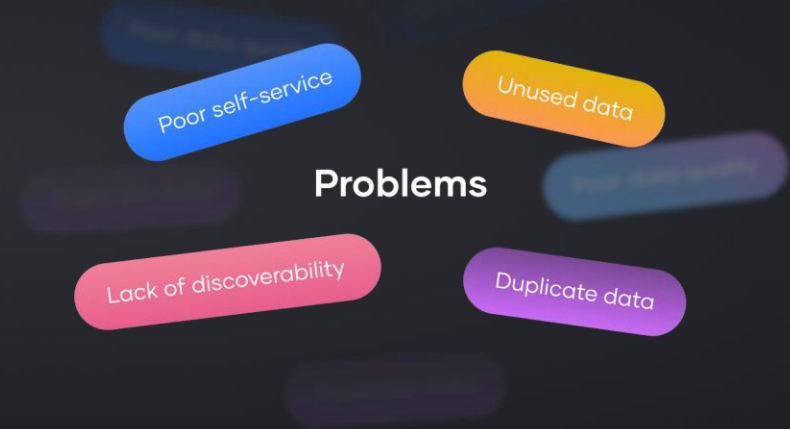


So, what can it do for you?



Problems related to lack of data modeling

- Duplicate or unused data
- Multiple names for common objects
- Poor self-service
- Lack of discoverability
- Meetings instead of documentation
- Poor data quality
- Business lack of trust in the data



“A lack of a data model is still a data model, albeit a crappy one.” - Joe Reis

Benefits associated with data modeling

- Consensus and visibility of the business model
- Reusability on subsequent projects and self-service
- Accelerated time to market and analytics
- Effective utilization of data assets unlocks their full value.
- Cross-platform, cross-domain, and universally understood
- Faster onboarding of new hires and consultants



TLDR;

Modeling saves time and time is money.

Our Differentiators



Ongoing Value

Receive monthly feature releases without the need to upgrade versions.



Anytime, anywhere

Access models from any location, and minimize downtime.



Empowering Teams

With our concurrent working decentralized teams can work collaboratively will focused on autonomous priorities. While following consistent standards and patterns.



Accelerate Design

Accelerate your time to market leveraging essential features such as naming standards, column templates and table templates

What does data modeling consist of?

Business model

What we do?

Data model

What data about what we do, do we care about?

Vocabulary

How do we “encode” these things for effective communication?

For example

Business model

We sell services to subscriptions

Data model

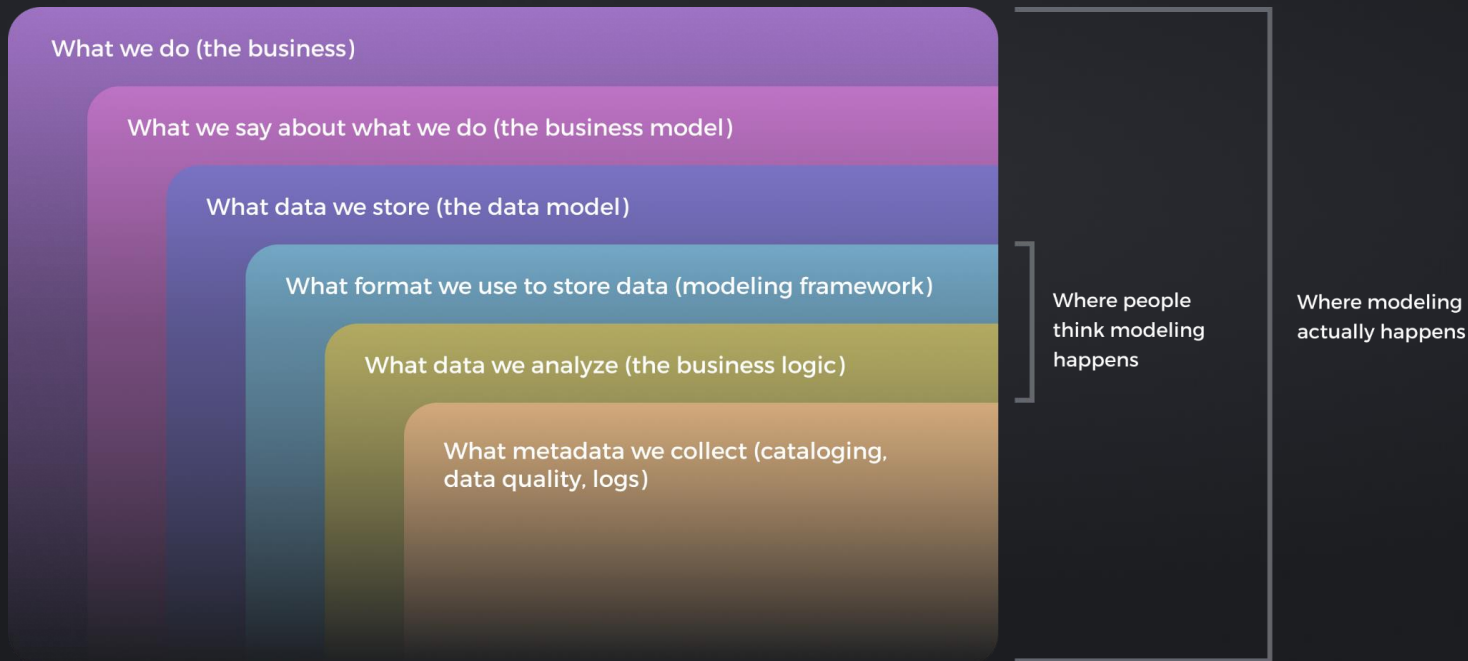
| Concepts | Collateral (<i>Ontology</i>) |
|-----------------------------------|--|
| Service, service type, subscriber | Entity, Relationship, ERD, Data Dictionary |

Vocabulary

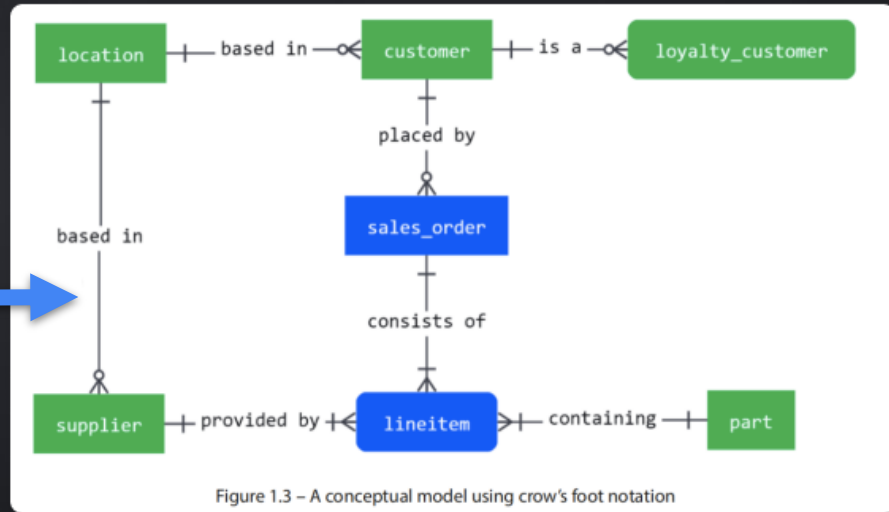
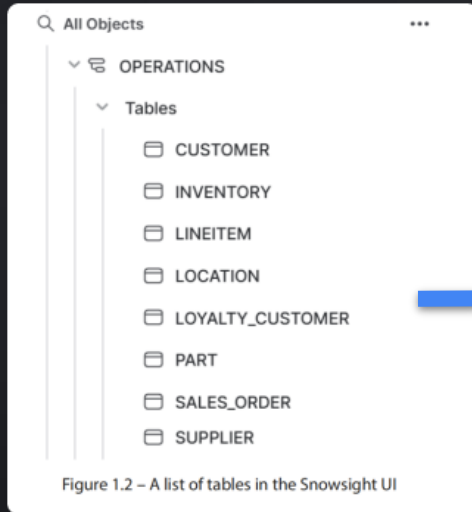
Active subscriber, type 2 dim, normalization, logical deletion

Modeling is a team sport

Getting buy-in from business teams and product teams

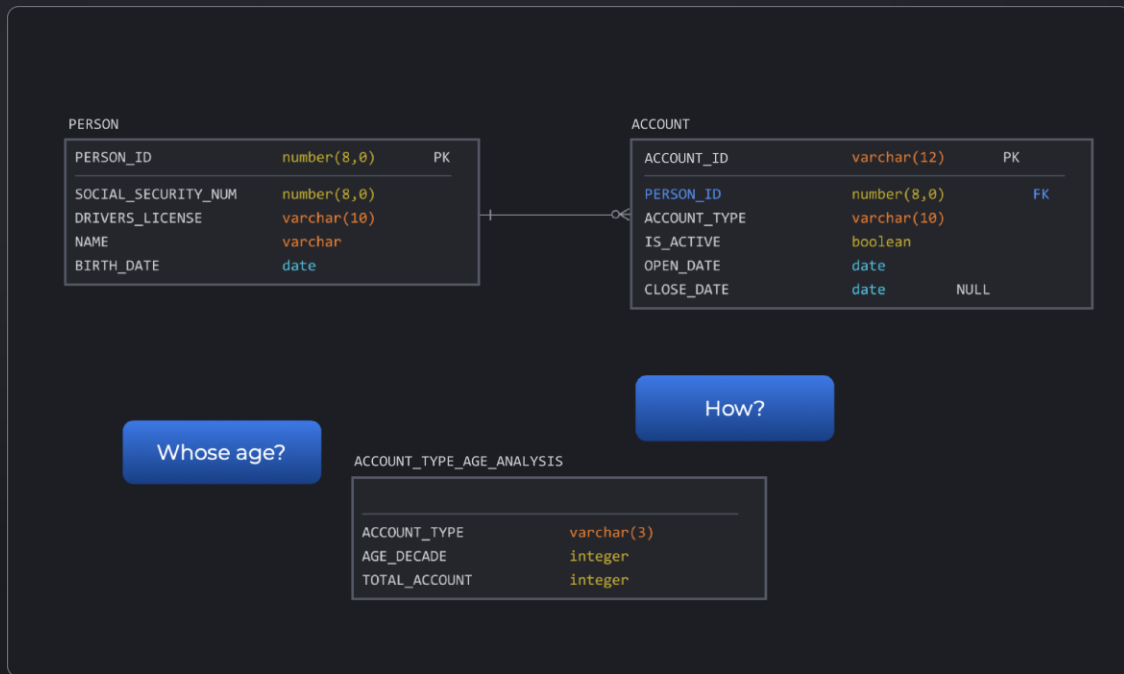


Leveraging and delegating data modeling know-how in the organization



“There needs but one wise man in a company and all are wise, so rapid is the contagion.” - Ralph Waldo Emerson

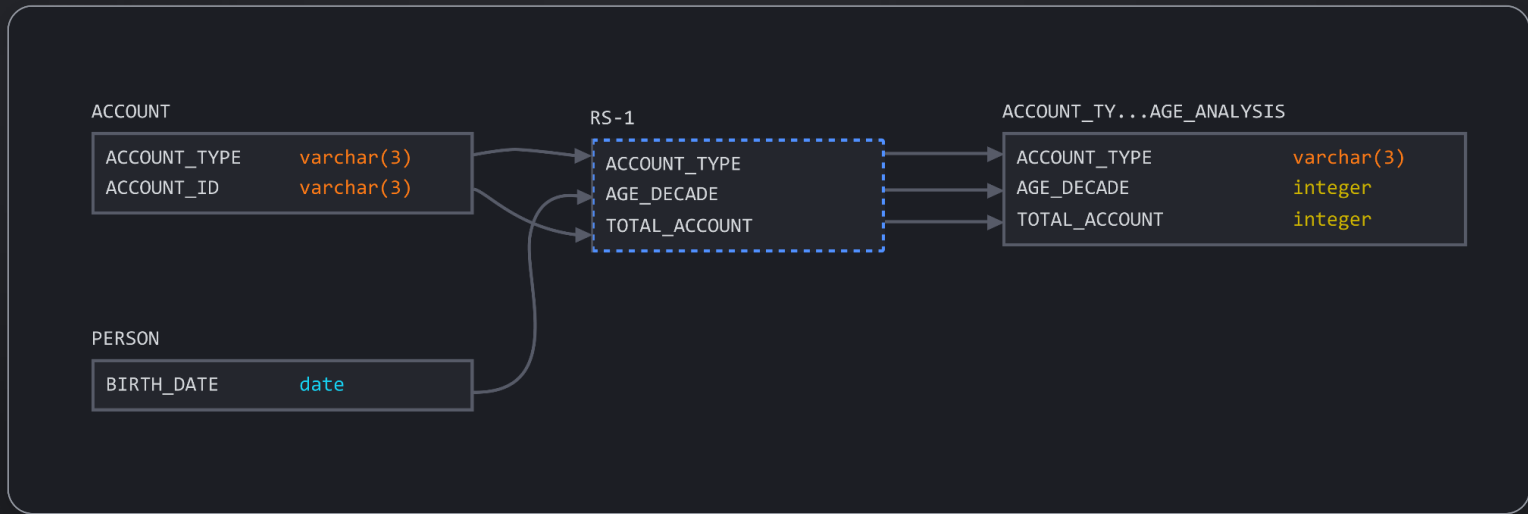
Relational vs transformational modeling (SqlDBM vs dbt)



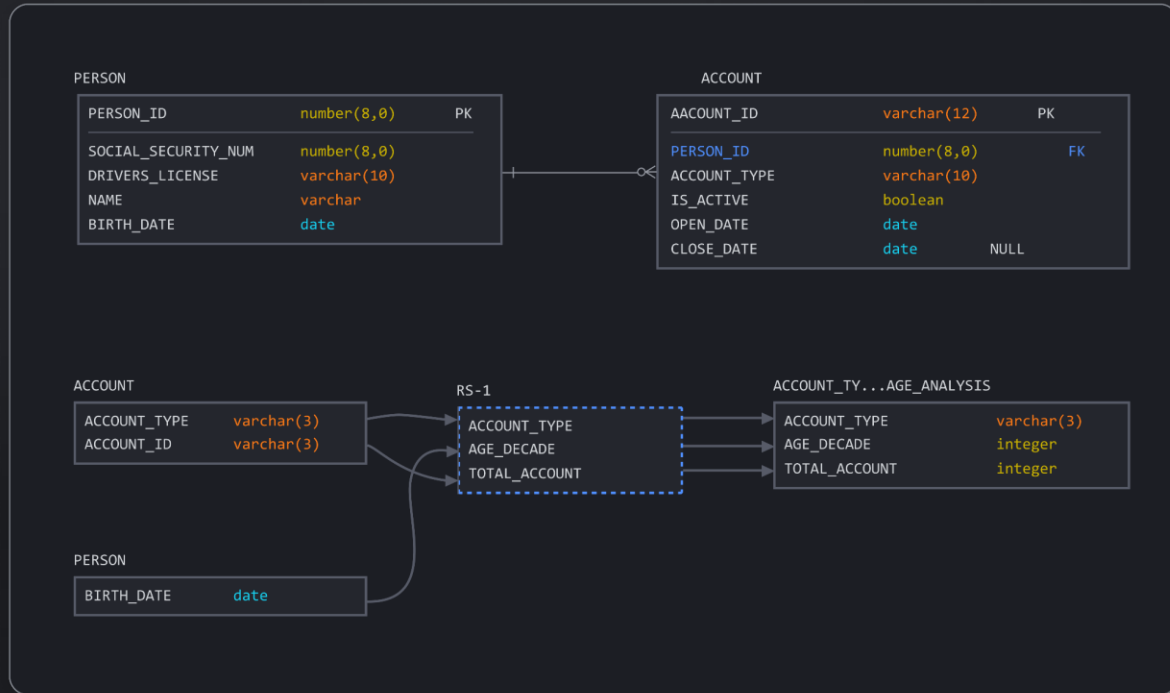
Relational vs transformational modeling (SqlDBM vs dbt)

```
CREATE TABLE account_types_age_analysis AS
SELECT
  a.account_type,
  ROUND(DATEDIFF(years, p.birth_date, CURRENT_DATE()), -1
) AS age_decade,
COUNT(a.account_id) AS total_accounts
FROM account AS a
INNER JOIN person AS p
ON a.person_id = p.person_id
GROUP BY 1, 2;
```

Relational vs transformational modeling (SqlDBM vs dbt)



Relational vs transformational modeling (SqlIDBM + dbt)

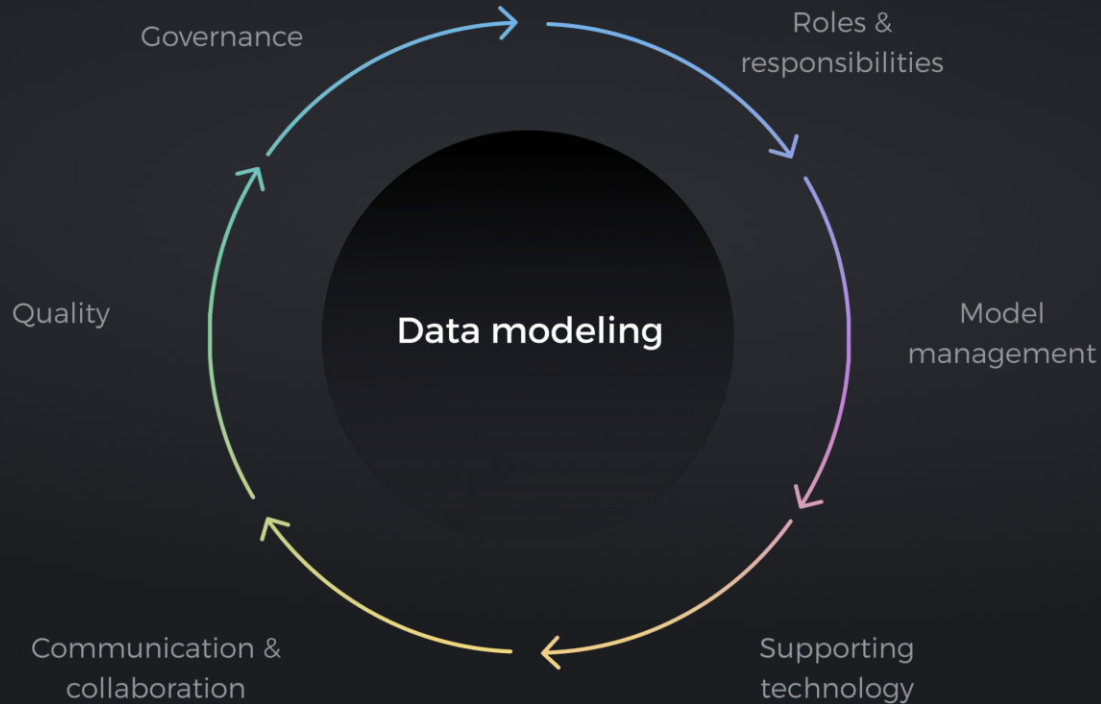


Establishing a data modeling framework

- Nominate a champion
- Obtain buy-in
- Create a process
- Demonstrate value early
- Democratize it (make it accessible to everyone)



Data modeling framework



Data modeling framework



Roles & Responsibilities

Define the key roles involved in data modeling within the organization, such as Data Architects, Data Modelers, and Data Stewards, and outlines their specific responsibilities and contributions to the data modeling process, ensuring clear accountability and effective collaboration



Model Management

Establishing a structured process for the lifecycle of data models, from conception to deployment. It encompasses practices for ensuring consistency, compliance with business requirements, and effective version control, thereby enhancing the integrity and usability of data models.



Supporting Technology

Identifying and standardizing the tools and technologies used in data modeling, such as specific modeling software and version control systems. It's about leveraging technology to improve efficiency, ensure consistency, and support the various stages of data modeling.

Data modeling framework



Communication & Collaboration

Emphasize the importance of clear communication and collaborative practices in data modeling. It includes strategies for sharing information, fostering teamwork across different roles, and ensuring that data modeling efforts are aligned with organizational goals and integrated with other business processes.



Quality

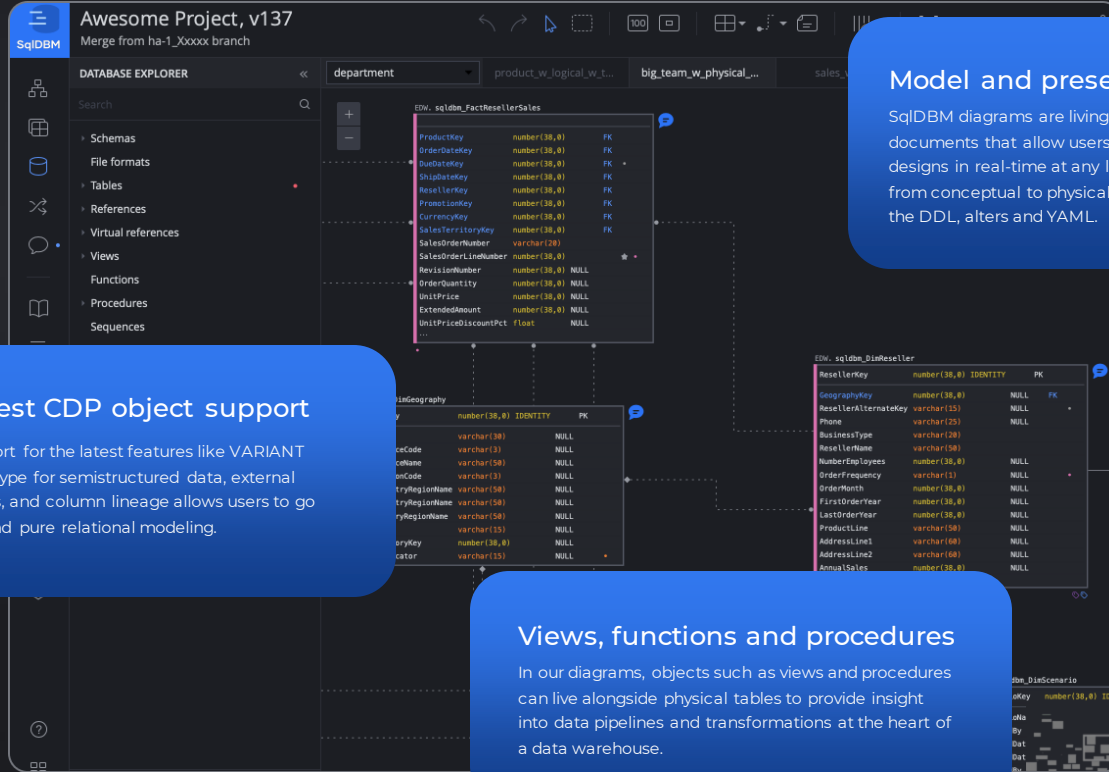
Maintaining high standards of quality in data models. It involves setting quality benchmarks, regular reviews and audits of data models, and implementing feedback mechanisms to continuously improve the accuracy, efficiency, and effectiveness of the data modeling process.



Governance

Establishment of policies and standards for data naming, metadata management, and model architecture. It ensures that data models adhere to organizational standards, legal compliance, and best practices, thereby safeguarding data integrity and facilitating effective data management.

Let's talk about the product



Model and present

SqlDBM diagrams are living interactive documents that allow users to present their designs in real-time at any level of detail - from conceptual to physical - and generate the DDL, alters and YAML.

Latest CDP object support

Support for the latest features like VARIANT data type for semistructured data, external tables, and column lineage allows users to go beyond pure relational modeling.

Views, functions and procedures

In our diagrams, objects such as views and procedures can live alongside physical tables to provide insight into data pipelines and transformations at the heart of a data warehouse.

SqlDBM basics

Unlimited projects, unlimited change history, unlimited objects

Awesome Project, v137
Merge from ha-1_XXXXX branch

FORWARD ENGINEER

```
----- SqlDBM: Snowflake -----  
----- Generated by SqlDBM: Awesome Project by yana.danilovich@sqldb.com -----  
-----  
----- EDW.cos_center -----  
CREATE TAG EDW.cos_center.ALLOWED_VALUES 'val_1', 'val_2'  
COMMENT = 'cost center tag';  
-----  
----- TAG_SCHEMA_Issues -----  
CREATE TAG_TAG_SCHEMA_Issues;  
-----  
----- TAG_SCHEMA_Medallion -----  
CREATE TAG_TAG_SCHEMA_Medallion.ALLOWED_VALUES 'Gold', 'Silver', 'Bronze';  
-----  
----- HOTELBNB_Top_2 -----  
CREATE TAG_HOTELBNB_Top_2.ALLOWED_VALUES 'val_1'  
COMMENT = 'demo';  
-----  
----- sqldb_table_w_warning -----  
CREATE TABLE sqldb_table_w_warning  
(  
);  
-----  
----- EDW.sqldb_DimSalesTerritory -----  
CREATE TABLE EDW.sqldb_DimSalesTerritory  
(  
SalesTerritoryKey number(38,0) NOT NULL AUTOINCREMENT START 1 INCREMENT 1 COMMENT 'Sales Territory Identit  
SalesTerritoryAlternateKey number(38,0) COMMENT 'Sales Territory Alternate Key',  
SalesTerritoryRegion varchar(50) NOT NULL COMMENT 'Sales Territory Region',  
SalesTerritoryCountry varchar(50) NOT NULL COMMENT 'Sales Territory Country',  
SalesTerritoryGroup varchar(50) COMMENT 'Sales Territory Group',  
CONSTRAINT PK_DimSalesTerritory_sqldb_DimSalesTerritory PRIMARY KEY (SalesTerritoryKey )  
COMMENT = 'Sales Territory data';  
-----  
----- EDW.sqldb_DimPromotion -----  
CREATE TABLE EDW.sqldb_DimPromotion  
(  
PromotionKey number(38,0) NOT NULL AUTOINCREMENT START 1 INCREMENT 1 COMMENT 'Promotion identifier',  
PromotionAlternateKey number(38,0) COMMENT 'Promotion alternate key',  
EnglishPromotionName varchar(255) COMMENT 'Promotion name in English',  
SpanishPromotionName varchar(255) COMMENT 'Promotion name in Spanish',  
FrenchPromotionName varchar(255) COMMENT 'Promotion name in French',  
DiscountPct float COMMENT 'The percentage of the discount',  
EnglishPromotionType varchar(50) COMMENT 'Promotion type in English',  
SpanishPromotionType varchar(50) COMMENT 'Promotion type in Spanish',  
);  
-----
```

GENERATION OPTIONS

Repository folder

Migrations

Source DDL files

DIAGRAM PROPERTIES

Diagram area

ProductKey
Lookup

Department
big_team_w_physical_w_flag

View mode options

Show schema Show relationships

Relationship logical name

Data type Null option Keys

Virtual keys Color data type

Fixed object height

Show many-to-many as diamond

Global user preferences

Position

Description

Diagram with a big physical model, flags and role name

Object flow

| Table | Columns | Types | Constraints |
|-----------------------------|----------------------|--------------|-------------|
| department | departmentid | number(38,0) | PK |
| product_w_logical_w_... | productcategoryid | number(38,0) | FK |
| product_w_logical_w_... | productsubcategoryid | number(38,0) | FK |
| product_w_logical_w_... | productid | number(38,0) | FK |
| big_team_w_physical_w_... | departmentid | number(38,0) | FK |
| big_team_w_physical_w_... | productid | number(38,0) | FK |
| big_team_w_physical_w_... | productcategoryid | number(38,0) | FK |
| big_team_w_physical_w_... | productsubcategoryid | number(38,0) | FK |
| EDW.sqldb_FactInternetSales | productid | number(38,0) | FK |
| EDW.sqldb_FactInternetSales | productcategoryid | number(38,0) | FK |
| EDW.sqldb_FactInternetSales | productsubcategoryid | number(38,0) | FK |
| EDW.sqldb_FactInternetSales | productid | number(38,0) | FK |
| EDW.sqldb_FactInternetSales | customerid | number(38,0) | FK |
| EDW.sqldb_FactInternetSales | promotionid | number(38,0) | FK |
| EDW.sqldb_FactInternetSales | currencyid | number(38,0) | FK |
| EDW.sqldb_FactInternetSales | salesorderid | number(38,0) | FK |
| EDW.sqldb_FactInternetSales | salesorderlineitemid | number(38,0) | FK |
| EDW.sqldb_FactInternetSales | revenue | number(38,0) | |
| EDW.sqldb_FactInternetSales | drainquantity | number(38,0) | |
| EDW.sqldb_FactInternetSales | dispprice | number(38,0) | |
| EDW.sqldb_FactInternetSales | extradiscount | number(38,0) | |
| EDW.sqldb_FactInternetSales | discountpct | float | |
| EDW.sqldb_FactInternetSales | discountamount | float | |

- No limit on number of objects or revisions
- Each diagram supports 3000+ objects
- Web-based/ SaaS
- Project-based

Let's talk about the product



Relational modeling



Model-driven collaboration, governance, and self-service

✓ Visualize

✓ Relate

✓ Search

✓ Design

✓ Filter

✓ Deploy

✓ Govern

✓ Monitor

✓ Discover

✓ Integrate

Column view lineage

Break down complex data flows with column-level lineage and substep visualization. Take advantage of syntax highlighting from the diagram as well as code editor to see even real-time changes.

The screenshot displays the SQLDBM interface for an "AdWorks project, v652" (sprint 7 2022). The main window is titled "COLUMN LINEAGE ANALYSIS" and shows a detailed view of the "LOYALTY_V" view. The left sidebar lists several views, with "LOYALTY_V" selected. The central area contains a complex lineage diagram with nodes representing tables and views, connected by lines indicating data flow. The bottom panel shows the SQL code for the view, with syntax highlighting and a "Visualize all views" button.

```
SQL code editor content:  
35 FROM SQLDBM_DEMO_SF1K_DATA.ORDERS  
36 )  
37 )  
38 , cust_ord as (  
39 )  
40 select o_custkey, sum(o_totalprice) as o_totalprice from (  
41 select o.*, c.*  
42 from ord o  
43 inner join cust c  
44 on o.o_custkey = c_custkey  
45 where true  
46 and c_acctbal > 0 --no deadbeats  
47 and c_nationkey != 22 -- Excluding Russia from loyalty program will send strong message to Putin  
48 )  
49 group by 1
```


Main Menu

Use Main Menu to switch between other projects (Dashboard), team management (Project-Team), and App theme settings. You also can convert your project to other DB/DW type.

Add Objects

Click to Add Table, Relationship, or a Note. Objects can also be added by right-clicking on the diagram canvas or through keyboard shortcuts.

Export

Export the diagram to PNG or SVG.

Team Collaboration

Invite team members to collaborate on a project. Grant view/edit permissions

Object Explorer

Switch to see all diagrams and subject areas (Diagrams tab), objects of diagram (Diagram explorer tab) or objects of database (Database Explorer tab).

Database Documentation

Compare Revisions

Forward Engineer

Reverse Engineer

DataOps

Databricks Sample Project, v16

Merge from le-1 branch

Level_of_Detail Logical Physical Table PK/AK Keys TABLE PROPERTIES

| mySchema.orders | | | |
|-----------------|----------------|--|------|
| orderkey | bigint | | PK |
| custkey | bigint | | FK |
| orderstatus | string | | NULL |
| totalprice | decimal(18, 2) | | NULL |
| orderdate | date | | NULL |
| orderpriority | string | | NULL |
| clerk | string | | NULL |
| shippriority | int | | NULL |
| comment | string | | NULL |

| mySchema.Lineitem | | | |
|-------------------|----------------|--|----------|
| linenumber | int | | PK |
| orderkey | bigint | | PK FK |
| partkey | bigint | | NULL VFK |
| suppkey | bigint | | FK |
| quantity | decimal(18, 2) | | NULL |
| extendedprice | decimal(18, 2) | | NULL |
| discount | decimal(18, 2) | | NULL |
| tax | decimal(18, 2) | | NULL |
| returnflag | string | | NULL |
| linestatus | string | | NULL |
| shipdate | date | | NULL |
| commitdate | date | | NULL |
| receiptdate | date | | NULL |
| shipinstruct | string | | NULL |
| comment | string | | NULL |
| ETL_ID | timestamp | | |
| ETL_TIMESTAMP | timestamp | | |

| mySchema.part | | | |
|---------------|----------------|--|----------|
| partkey | bigint | | NULL VPK |
| name | string | | NULL |
| afgr | string | | NULL |
| brand | string | | NULL |
| type | string | | NULL |
| size | int | | NULL |
| container | string | | NULL |
| retailprice | decimal(18, 2) | | NULL |
| comment | string | | NULL |

| mySchema.partsupp | | | |
|-------------------|----------------|--|--------|
| partkey | bigint | | PK VFK |
| suppkey | bigint | | PK VFK |
| availability | int | | NULL |
| supplycost | decimal(18, 2) | | NULL |

Action Buttons

Switch between viewing and editing the project

Table Properties

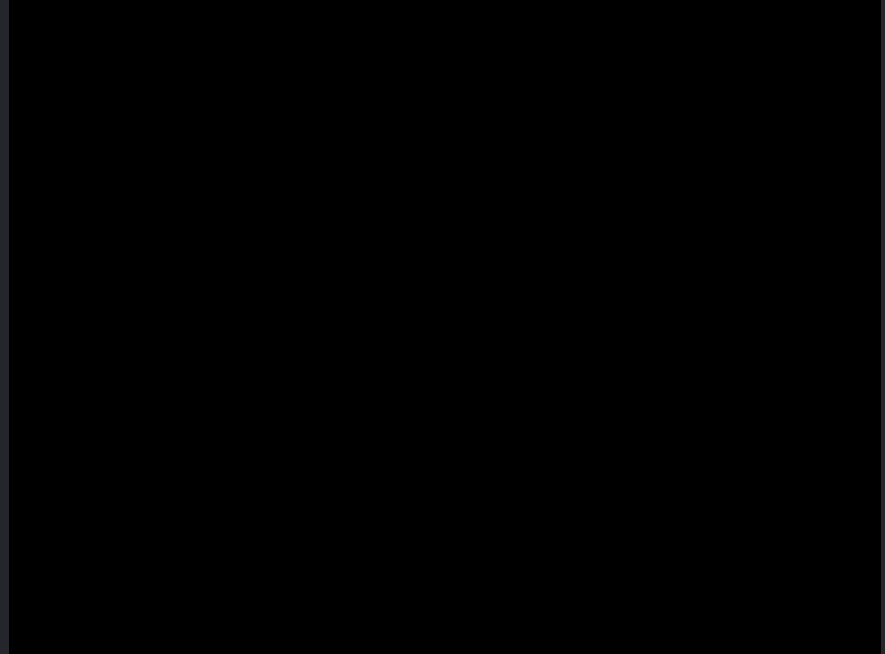
Select a table and expand tabs for editing columns, indexes, constraints, description, color indicator and related tables. Click on a column name to access Column Properties.

| mySchema.customer | | | |
|-------------------|--------|--|----------|
| custkey | bigint | | PK |
| name | string | | NULL |
| address | string | | NULL |
| Mainbranch | bigint | | NULL VFK |

| Columns | | | |
|---------------|----------------|--|-------|
| linenumber | int | | PK |
| orderkey | bigint | | PK FK |
| partkey | bigint | | VFK |
| suppkey | bigint | | FK |
| quantity | decimal(18, 2) | | |
| extendedprice | decimal(18, 2) | | |
| discount | decimal(18, 2) | | |
| tax | decimal(18, 2) | | |
| returnflag | string | | |
| linestatus | string | | |
| shipdate | date | | |
| commitdate | date | | |
| receiptdate | date | | |
| shipinstruct | string | | |
| shipmode | string | | |
| comment | string | | |
| ETL_ID | timestamp | | |
| ETL_TIMESTAMP | timestamp | | |

How to rely on standards and templates to keep things organized

- Establishing consistency
- SqlDBM conventions
 - Case
 - Object
- Column templates
- Table templates



Discoverability

The image displays the sqlDBM interface with three overlapping windows. The top window, titled 'Awesome Project, v137', shows the 'DATABASE DOCUMENTATION' view for the 'SFLK_DATA' schema, listing tables like 'SFLK_DATA.sqldbm_DimReseller' and 'ResellerKey'. The bottom-left window, titled 'Adventure Works, v77', shows the 'DATABASE EXPLORER' view with a tree structure of schemas and tables, including 'DimAccount'. The bottom-right window shows the 'Columns' view for 'EDW.sqldbm_DimC', displaying a table of columns with their names, descriptions, and flags.

| All | Object name | Column name | Description | Field | Flags | Tags |
|-----|------------------------|-------------|-------------|-------|-------|------|
| | EDW.sqldbm_44 | | | | | |
| | EDW.sqldbm_44 | | | | | |
| | EDW.sqldbm_44_1 | | | | | |
| | EDW.sqldbm_44_1 | | | | | |
| | EDW.sqldbm_ChildDemo | | | | | |
| | EDW.sqldbm_DimAccount | | | | | |
| | EDW.sqldbm_DimCurrency | | | | | |
| | EDW.sqldbm_DimCustomer | | | | | |

Documentation and governance

Enhancing data documentation features for successful data governance framework



Governance
Project Role



Custom fields



Pages



Flag and metadata
search



DB Documentation
Comments

Custom fields

Fields are additional metadata columns that can be created and maintained by users with the Governance role to provide additional attributes (e.g., owner, is validated, source) to objects within the project.

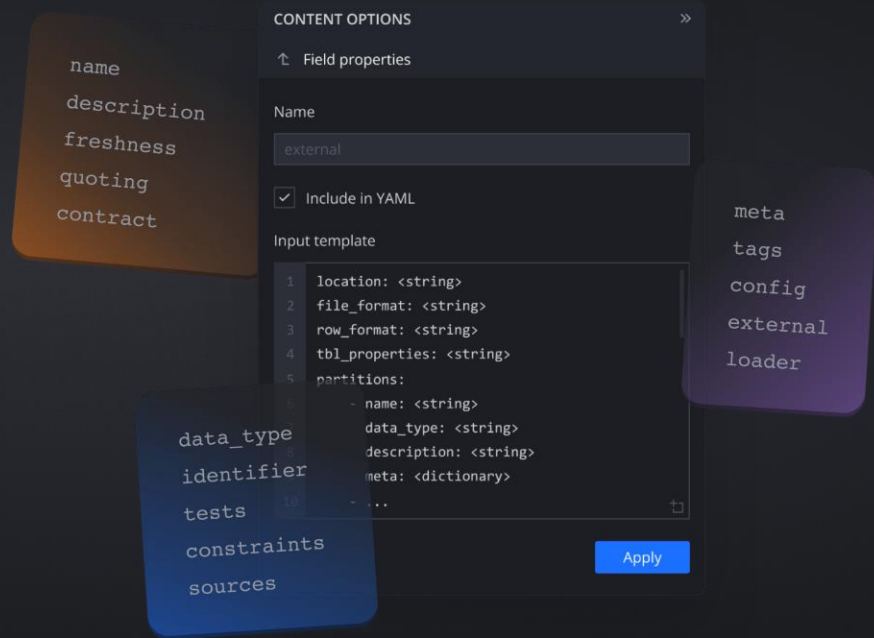
The screenshot displays the Google BigQuery interface for database documentation. The main table lists various objects and their associated custom fields. The table has columns for Name, Field_2, and Field_3. The objects listed include a Dataset and two Tables, each with multiple rows representing different levels of hierarchy. Custom fields are shown as dropdown menus with selected values.

| Name | Field_2 | Field_3 |
|---------------------------------|---------|----------------|
| Dataset : pasha_test | beta | beta |
| Table : pasha_test.co_table | dev | stage beta dev |
| a | dev | dev beta |
| a.a | dev | be stage |
| a_1 | dev | beta |
| Table : pasha_test.deep_sstruct | dev | beta dev |
| a1 | dev | stage |
| a1.b1 | dev | beta |
| a1.b1.c1 | dev | |
| a1.b1.c1.d1 | dev | dev beta stage |
| a1.b1.c1.d2 | dev | |
| a1.b1.c2 | dev | |
| a1.b2 | dev | |

On the right side, the 'DB DOCUMENTATION PROPERTIES' panel is visible, showing 'Content options' and 'Data governance fields'. Under 'Data governance fields', 'Field_2', 'Field_3', and 'Field_1' are listed with checkboxes. The 'Dbt enabled fields' section includes options like 'Identifier', 'loaded_at_field', 'tests', 'tags', 'freshness', 'quoting', 'external', 'quote', and 'docs'.

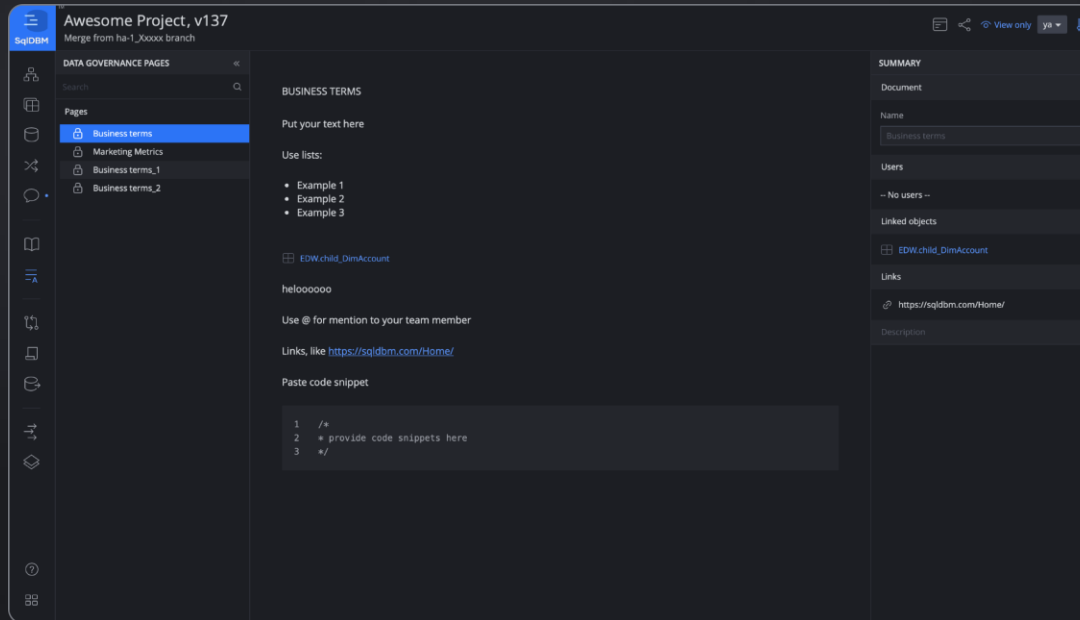
All dbt properties supported

Maintain all the supported dbt properties like tests and freshness at object or column level, all through pre-configured templates and visual interface.



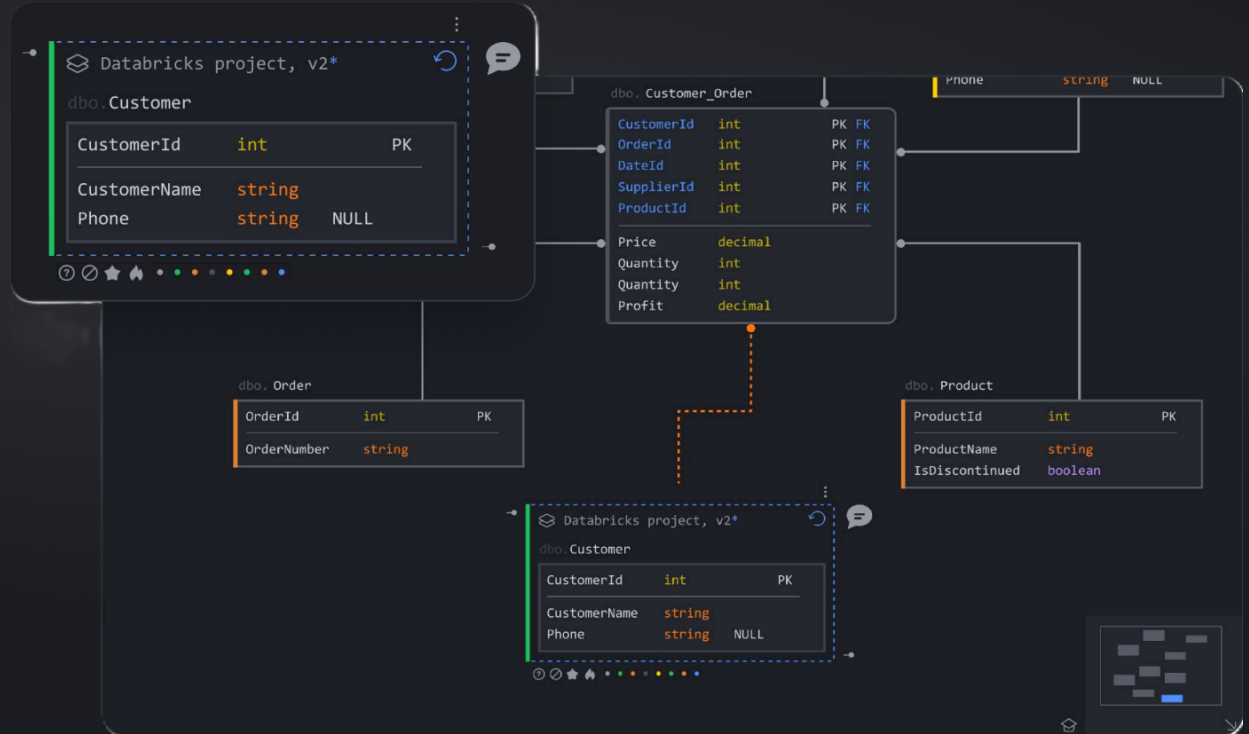
Pages

Extend documentation beyond the object level with rich text full-length content, embedded images, diagrams, styling, and formatting. Pages allow teams to maintain project-level overviews of things like metrics and KPI definitions, business terms, and operational outlines.



Global Modeling

Work across projects and business domains with object reference and change notification. Global modeling unlocks the potential of Data Mesh through SqlIDBM's all-in-one UI for search and discovery.



Forward engineering and CI/CD

The screenshot displays the SqLDBM interface for an 'AdWorks project' (v1027, sprint 23.36). The interface is divided into several sections:

- Project Overview (Left):** Shows the project name, sprint, and ticket number. It lists three environments: 'Prod' (Live DB revision), 'test env demo' (Upload DDL script), and 'Test' (Upload DDL script). Each environment has a 'Change revision' or 'Upload revision' button and a 'Se' (Select) button.
- Source Environment (Middle):** Titled 'SPRINT 23.36. TICKET NUMBER XYZ1234, V1027', it shows a list of objects with 'Added: 144 objects'. The selected object is 'EDW.DimScenario', which is highlighted in blue. Below the list, the SQL DDL script for this table is displayed.
- Target Environment (Right):** Titled 'PROD', it shows a list of objects with 'Removed: 3 objects' and 'Updated: 18 objects'. The selected object is also 'EDW.DimScenario', highlighted in blue. Below the list, the SQL DDL script for this table is displayed.

The diff view for the 'EDW.DimScenario' table shows the following SQL DDL scripts:

```
-- ***** SqLDBM: Snowflake *****  
-- ***** Generated by SqLDBM *****  
  
-- ***** EDW.DimScenario  
CREATE TABLE EDW.DimScenario  
(  
  ScenarioKey number(38,0) NOT NULL AUTOINCREMENT START 1 INCREMENT 1 COMMENT 'Scenario Ident',  
  ScenarioName varchar(50) COMMENT 'Name of the Scenario',  
  CreatedBy varchar(50) NOT NULL COMMENT 'Who Created this Scenario',  
  CreatedDate timestamp_ntz(9) NOT NULL COMMENT 'Scenario Creation Date',  
  UpdatedDate timestamp_ntz(9) NOT NULL,  
  UpdatedBy varchar(50) NOT NULL,  
  TestCol varchar(50) NOT NULL COMMENT 'comes from field ''customer_id''',  
  Col9 varbinary NOT NULL COMMENT 'isn't this great',  
  Col10 varbinary NOT NULL,  
  CONSTRAINT PK_DimScenario PRIMARY KEY ( ScenarioKey )  
)  
COMMENT = 'Dimension Scenario';
```

The PROD environment shows a similar script, but with a different primary key constraint:

```
-- ***** SqLDBM: Snowflake *****  
-- ***** Generated by SqLDBM *****  
  
-- ***** EDW.DimScenario  
CREATE TABLE EDW.DimScenario  
(  
  ScenarioKey number(38,0) NOT NULL AUTOINCREMENT START 1 INCREMENT 1 COMMENT 'Scenario Ident',  
  ScenarioName varchar(50) COMMENT 'Name of the Scenario',  
  CreatedBy varchar(50) NOT NULL COMMENT 'Who Created this Scenario',  
  CreatedDate timestamp_ntz(9) NOT NULL COMMENT 'Scenario Creation Date',  
  UpdatedDate timestamp_ntz(9) NOT NULL,  
  UpdatedBy varchar(50) NOT NULL,  
  TestCol varchar(50) NOT NULL,  
  CONSTRAINT PK_DimScenario PRIMARY KEY ( ScenarioKey )  
)  
COMMENT = 'Dimension Scenario';
```

Team communication

The screenshot displays the sqldb.com interface for a project titled "Awesome Project, v134". The main workspace shows a complex data model diagram with various tables and relationships. On the left, a "TEAM COMMENTS" sidebar lists recent comments from team members like Morris Will, Frank Nelson, and Francesca Mueller. On the right, a "DIAGRAM PROPERTIES" sidebar provides options for editing the diagram. A comment thread is overlaid on the right side of the diagram, showing a discussion about text replacement in a dialog box. The comment thread includes a header with the user's name and a timestamp, followed by the comment text and a "Reply" button.

Project Collaborators: Emily.Thomas@sqldb.com (Consumer)

Project Team: Cameron Williamson

Comment: @jurajc in the find & Replace dialog box of text documents, you can select to include the comments texts in your searches.

Comment: @linda.m Pls change

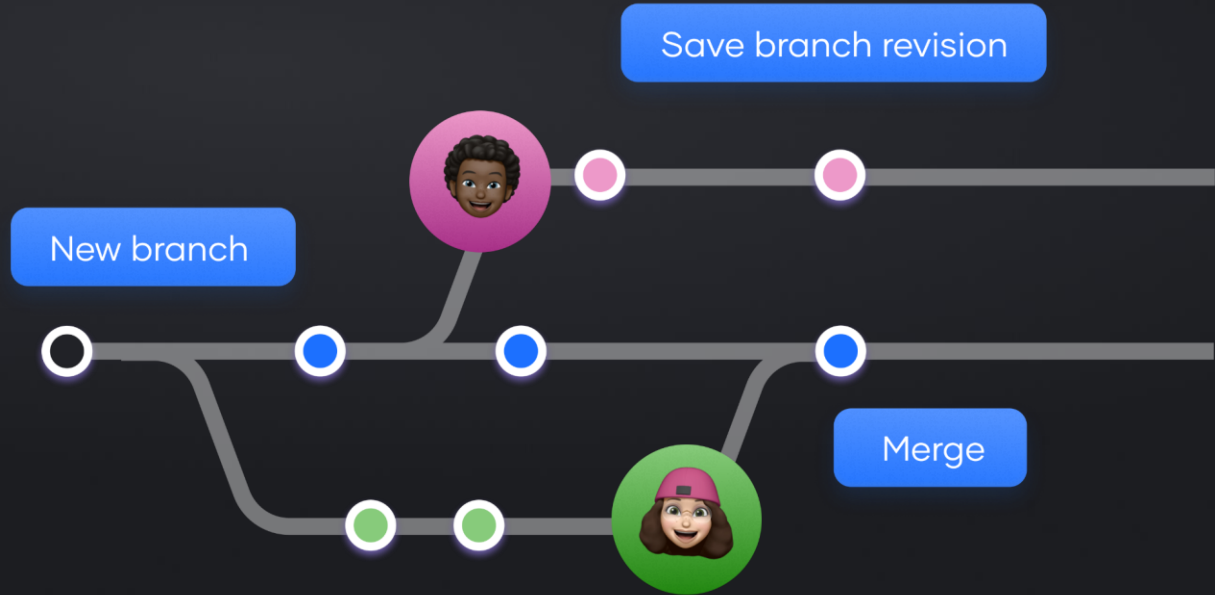
Comment: @linda.m Pls change

Comment: The draft text was submitted to the Venice Commission last year and comments on the text are expected soon.

Comment, resolve and close issues, use flagging to make your collaboration more effective.

Concurrent modelling

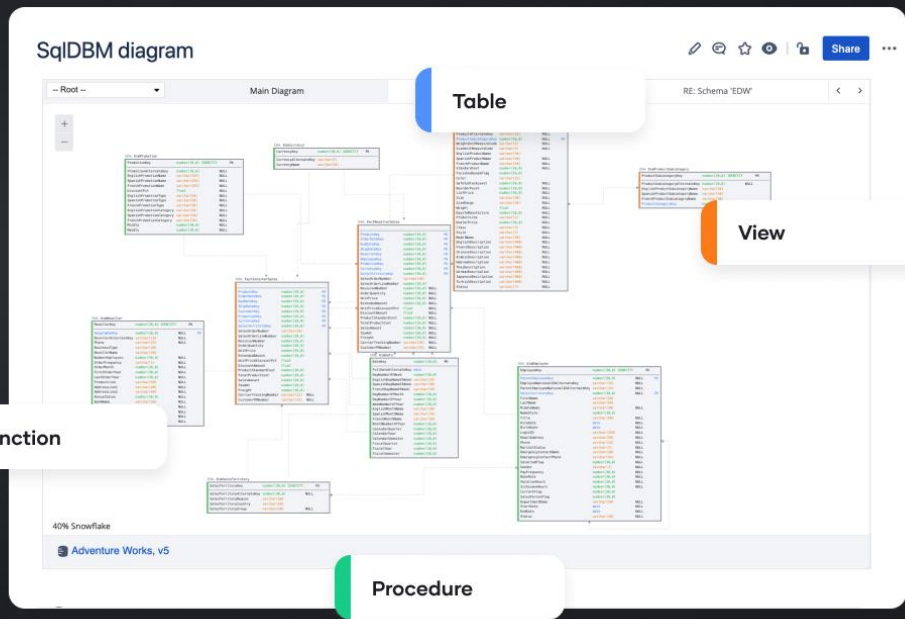
The entire project team can work collaboratively and in parallel without having to lock a project. Changes made in any branch can be compared before merging to main and synchronized with any new updates that happen meanwhile.





Share diagrams via Confluence

SqlDBM diagrams are now embeddable directly in Confluence. Share live, dynamic diagrams with your entire organization without needing to access SqlDBM or creating individual viewer accounts





Jira integration

Linking to related Jira tickets and issues ties project changes to the relevant business context and audit trails.

The integration will allow users to jump from a revision straight to a Jira ticket and vice versa - even from diagram comments!

The screenshot shows a Jira issue page for 'Fix Dim_product table' in the 'IN PROGRESS' status. The page includes a description, activity feed, and details panel. Annotations highlight the 'DONE' status, the 'IN PROGRESS' status, and the 'TO DO' status.

IN PROGRESS

Add epic / DBO-6

Fix Dim_product table

Attach Add a child issue Link issue

Description

DONE

Activity

Show: All Comments History Newest first

Add a comment...

Pro tip: press M to comment

SJ SqIDBM for Jira 2 minutes ago

Linked with SqIDBM: Snowflake SampleDb, v3 by John Smith

Fix Dim_product table fixed, please check

Edit Delete

IN REVIEW

IN PROGRESS

In Progress

Pinned fields

Click on the next to a field label to start pinning.

Details

Assignee Unassigned

Labels None

Reporter John Smith

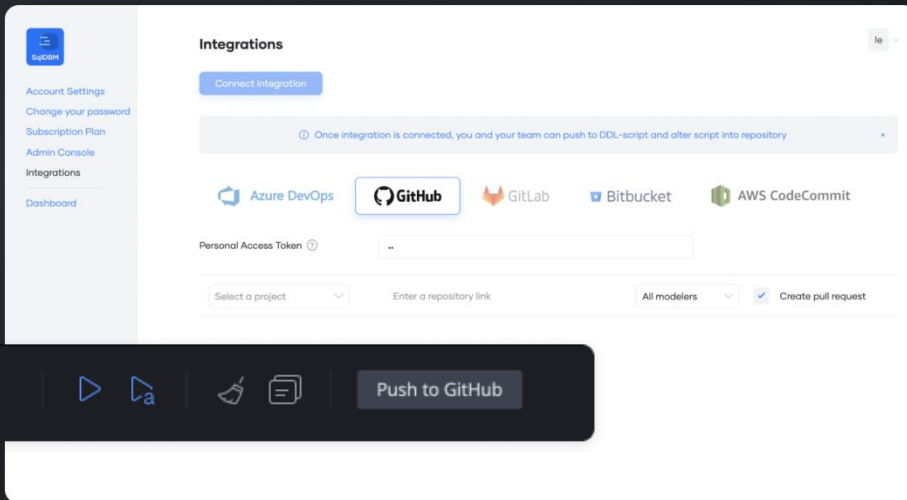
Created 18 minutes ago Updated 2 minutes ago Configure

TO DO



Materialize your changes and push to Git

Compare project revisions to live DB environments and generate ad-hoc alter scripts between them. Pair your project with any online git repository to automatically push alters or individual objects to new branches and pair with CI/CD tools like Schemachange or Flyway for fully automated deployments.



API

Interacting with your data model through a programmatic interface unlocks exciting options for data developers and engineers.

From sharing documentation with data catalogs to automating a CI/CD pipeline, APIs unlock a world of possibilities for engaging with SqIDBM models.

The screenshot shows the SqIDBM API Reference interface for the 'Get DDL' endpoint. The interface is dark-themed and includes a sidebar with navigation options, a main content area with the endpoint details, and a right-hand panel for request configuration and response examples.

Navigation: v1.0.4, API Reference, Changelog, Search, CTRL-K, Log In.

Endpoint: `GET https://api.sqidbm.com/projects/{projectId}/revisions/{revisionId}/ddl`

Methods: GET

Path Params: `projectId int32 required`, `revisionId int32 required`

Headers: `Accept string` (value: `application/xml`)

Request:

```
1 curl --request GET \
2 --url https://api.sqidbm.com/projects/{projectId}/revisions/{revisionId}/ddl \
3 --header 'accept: application/json'
```

Response: `application/json` (status: 200)

Footer: Powered by ReadMe

Get it right the first time!

1. Self-serve data discovery and relationship insights

2. Architect and validate high-level design with stakeholders

3. Data model becomes dbt contract in generated YAML

4. dbt contract enforces agreed-upon design in the dbt model

| Column Name | Column Type | Constraint |
|---------------|--------------|------------|
| loyalty_id | varchar | PK |
| customer_id | number(38,0) | FK |
| points_amount | number | |
| type | varchar | |
| comment | varchar | NULL |

```
models:
  - name: loyalty
    description: client loyalty program
    config:
      contract:
        enforced: true
    columns:
      - name: loyalty_id
        description: 'unique id: loyalty_id'
      - name: customer_id
        description: customer id
      - name: points_amount
        description: total points
      - name: type
```

| column_name | definition_type | contract_type | mismatch_reason |
|-------------|-----------------|-----------------------|-----------------------|
| COMMENT | TEXT | TEXT | data type mismatch |
| CUSTOMER_ID | FIXED | missing in definition | missing in definition |
| NEW_COL | TEXT | missing in contract | missing in contract |

Future Enhancements

