



W A L D

A Modern & Sustainable Analytics Stack

PyConDE / PyData 2023, April 17th

Florian Wilhelm



inovex



Dr. Florian Wilhelm

◆ inovex • HEAD OF DATA SCIENCE

- 🧠 Mathematical Modelling
- ↳ Modern Data Warehousing & Analytics
- 👤 Personalisation & RecSys
- 🎲 Uncertainty Quantification & Causality
- 🐍 Python Data Stack
- 🐱 OSS Contributor & Creator of PyScaffold

🐦 @FlorianWilhelm

🐱 Florian Wilhelm

🌐 FlorianWilhelm.info

✉️ florian.wilhelm@inovex.de





is an innovation and quality-driven IT project house with a focus on digital transformation.

- ▶ **Application Development** (Web Platforms, Mobile Apps, Smart Devices and Robotics, UI/UX design, Backend Services)
- ▶ **Data Management and Analytics** (Business Intelligence, Big Data, Searches, Data Science and Deep Learning, Machine Perception and Artificial Intelligence)
- ▶ **Scalable IT-Infrastructures** (IT Engineering, Cloud Services, DevOps, Replatforming, Security)
- ▶ **Training and Coaching** (inovex Academy)



**Using technology to
inspire our clients.
*And ourselves.***

www.inovex.de

Berlin · Karlsruhe · Pforzheim · Stuttgart · München · Köln · Hamburg · Erlangen

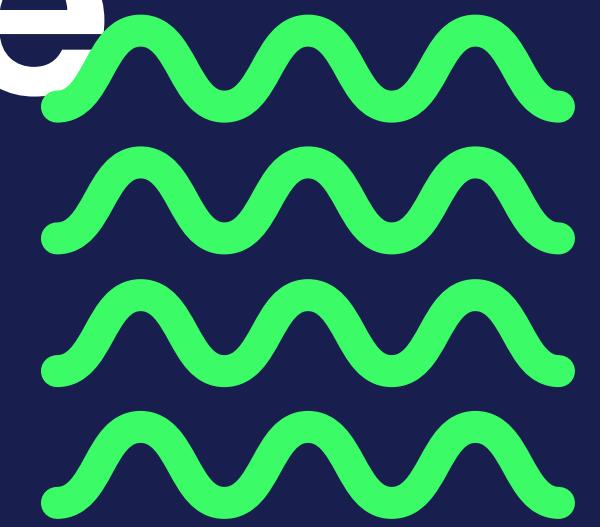


Data Warehouse

VS

Data Lake

&



ETL

VS

ELT



inovex

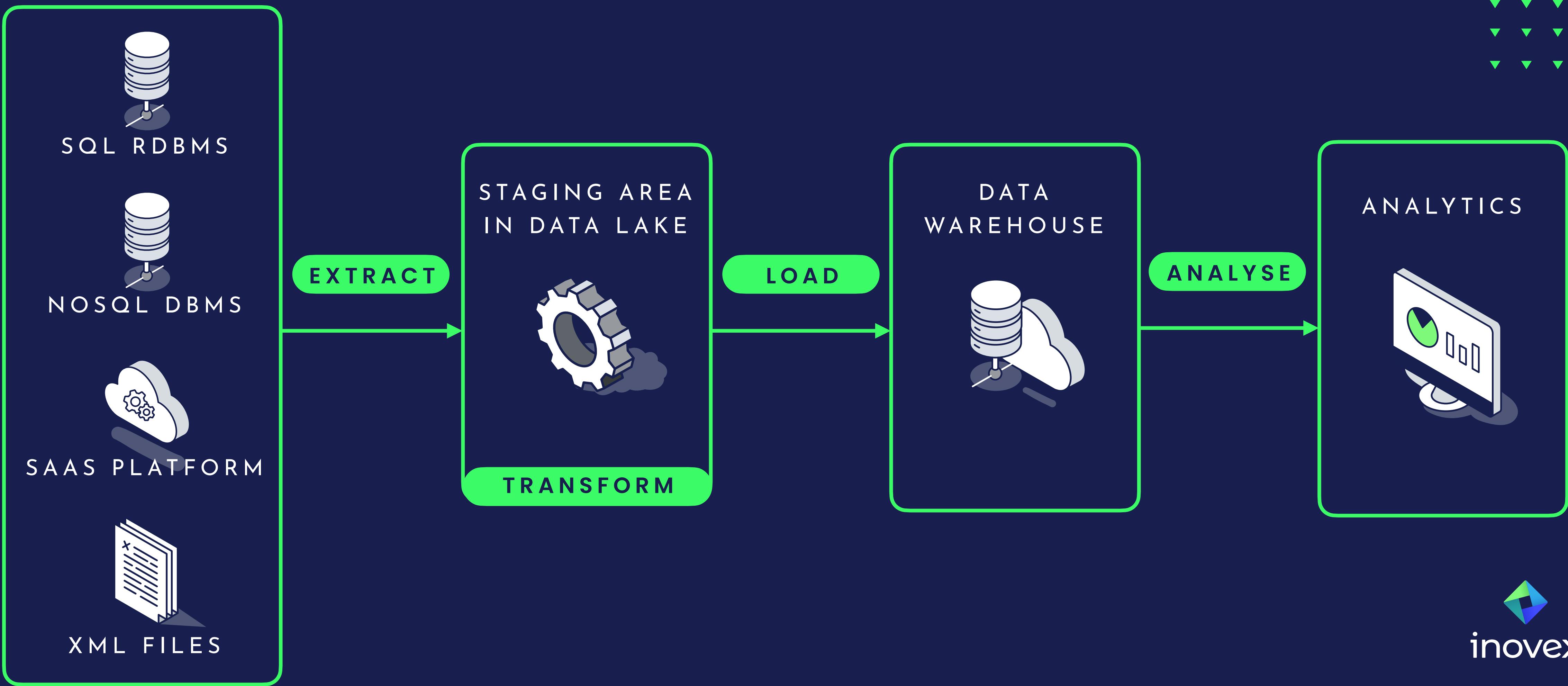
COMPARISON

Data Lake vs Data Warehouse

	Data Lake 	Data Warehouse 
Data Structure	Raw	Processed
Purpose of Data	Not yet determined	Currently in use
Users	Data scientists / engineers	Business professionals
Accessibility	Highly accessible and quick to update	More complicated and costly to make changes

COMPARISON

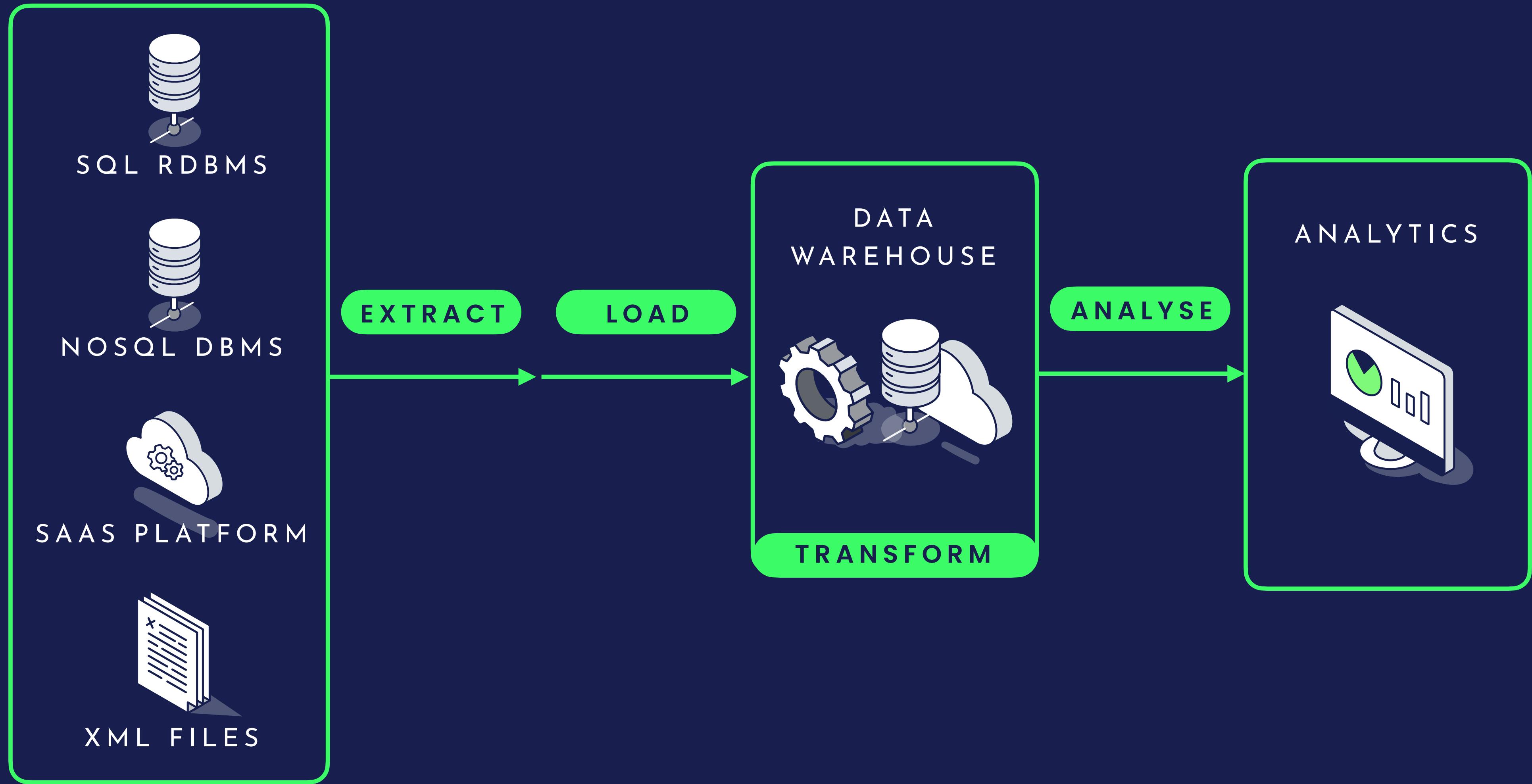
ETL – Extract, Transform, Load



inovex

COMPARISON

ELT – Extract, Load, Transform



PROGRESS

The Evolution of Databricks & Snowflake



databricks

DATA LAKE
& ETL



snowflake

DATA WAREHOUSE
& ELT



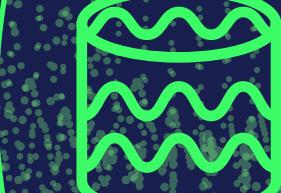
PROGRESS

The Evolution of Databricks & Snowflake

DATABRICKS
LAKEHOUSE

databricks

UNIFICATION
OF DATA WAREHOUSE & LAKE



SNOWFLAKE
DATA CLOUD

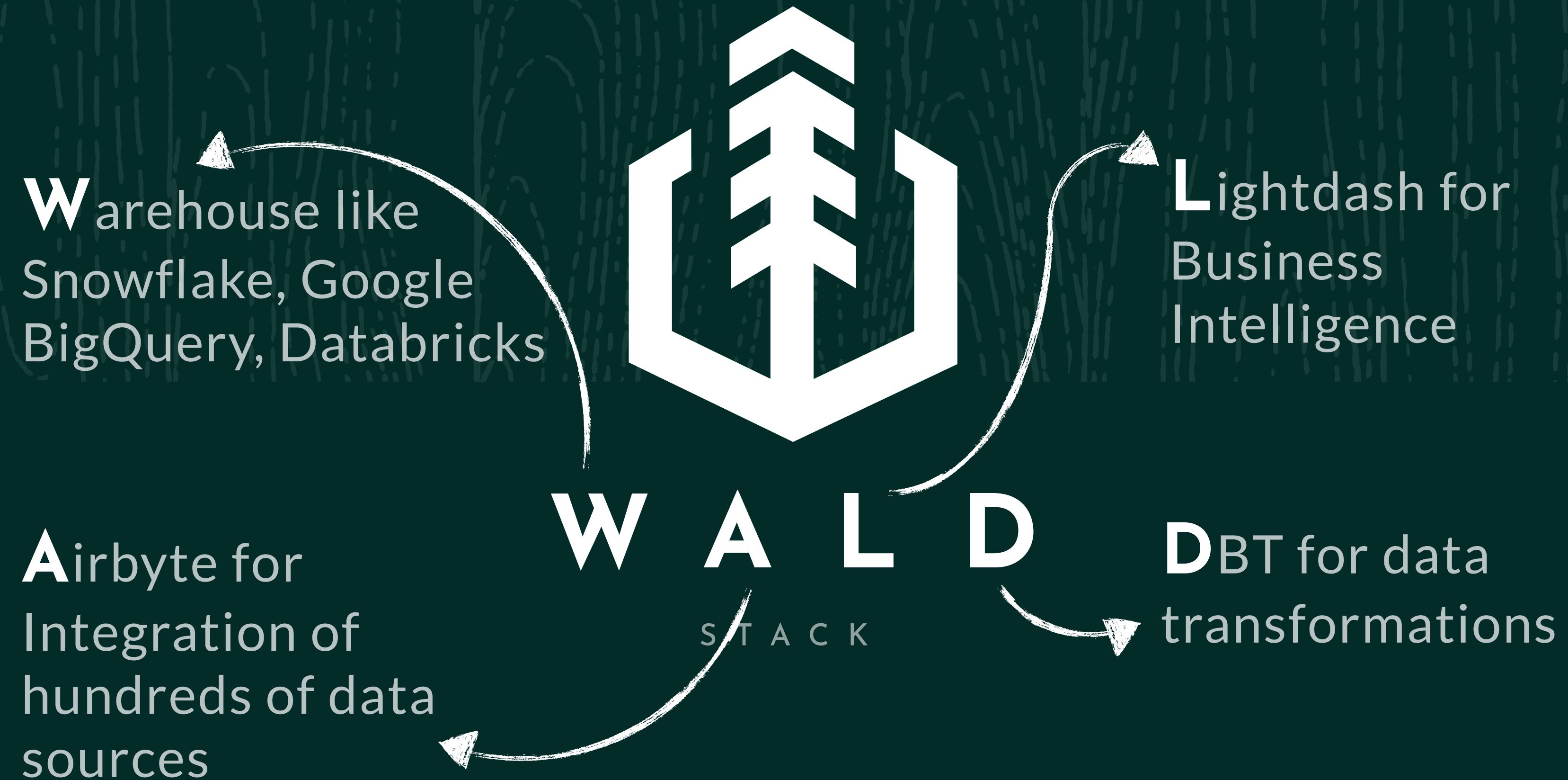


inovex





► A Modern & Sustainable Analytics Stack



WALD based on Google Big Query



- ▶ fully managed enterprise data warehouse
- ▶ only available on Google Cloud Platform
- ▶ uses dbt-bigquery to run Python UDFs with Dataproc using PySpark

WALD STACK

WALD based on Databricks



- ▶ uses Databrick's SQL warehouse
- ▶ available on all major cloud providers
- ▶ uses dbt-databricks to run Python UDFs with PySpark



WALD STACK

WALD based on Snowflake

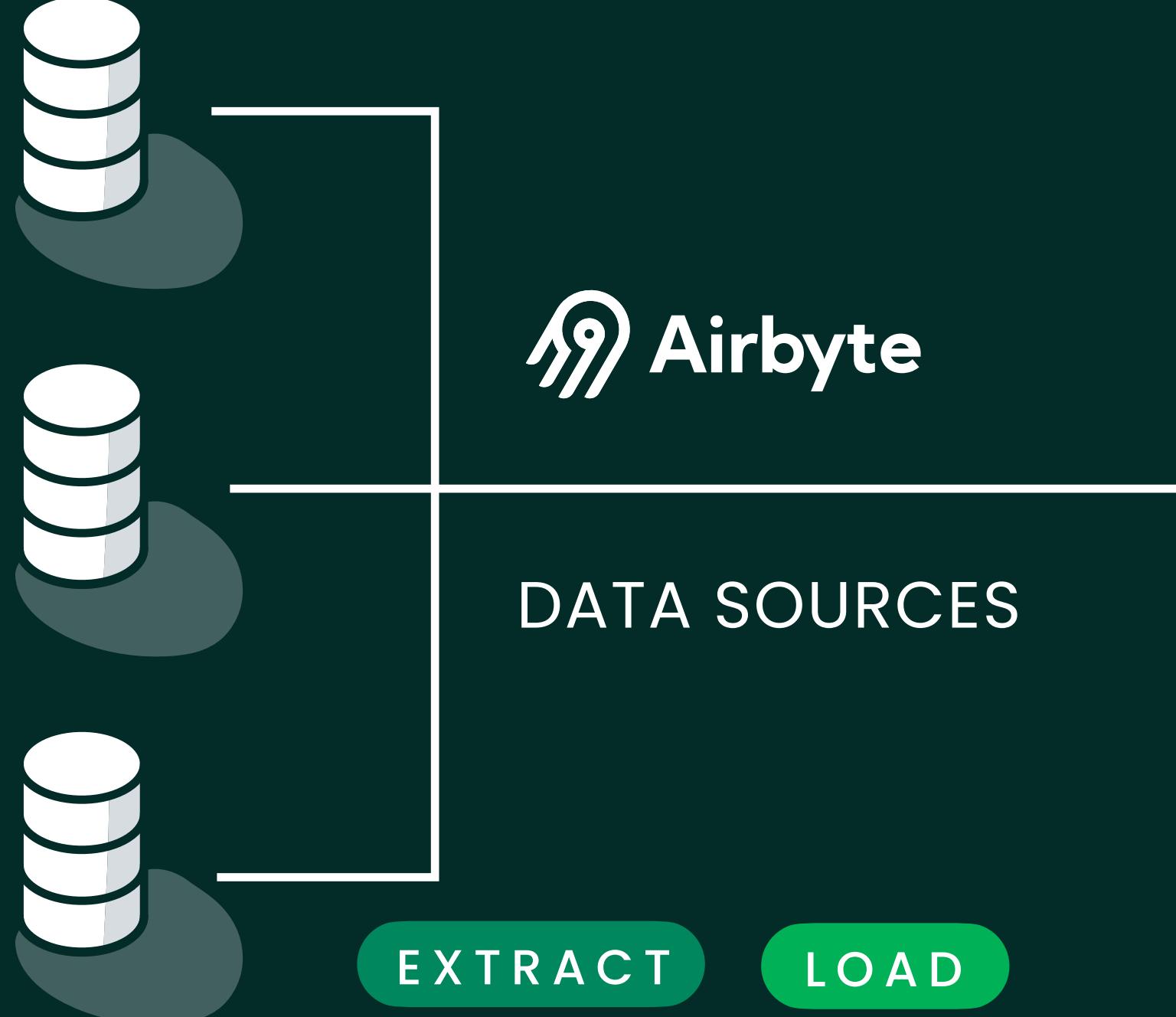


- ▶ comes with all the benefits of Snowflake, e.g. governance, easy collaboration, security, near zero maintenance
- ▶ runs on all major cloud providers
- ▶ tons of great documentation
- ▶ the easiest to use and full-featured option to get started with WALD



WALD STACK

Modern Analytics Stack



DATA CLOUD



TRANSFORM

SNOWPARK

 **Lightdash**





WALD STACK

Example Use-Case

find all code & details under

<https://waldstack.org>

FORMULA 1

Analysis and Position Prediction

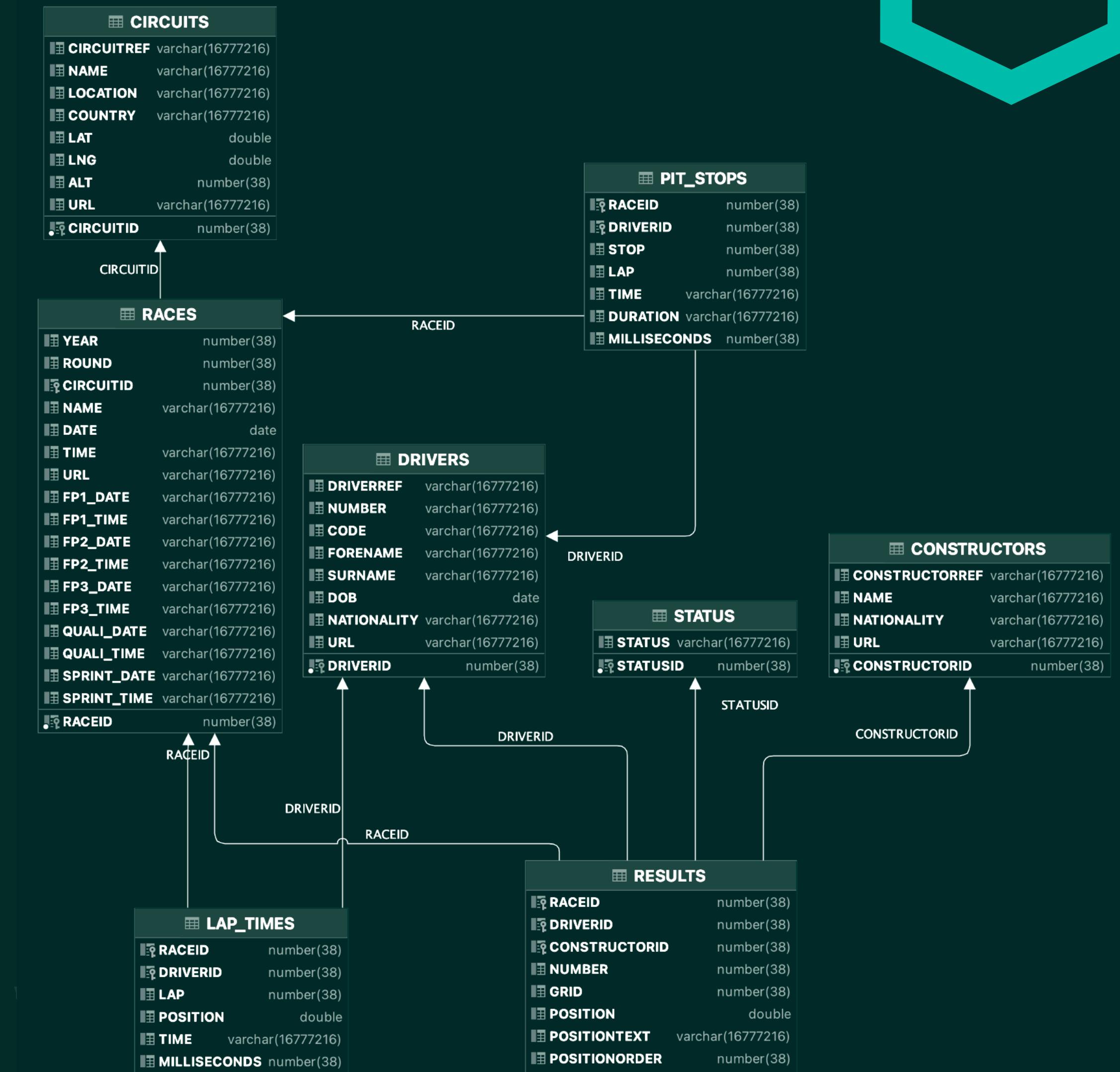
1. Upload data using Airbyte to Snowflake
2. Use dbt for some basic analysis
3. Use dbt with Snowpark for some ML to predict future position
4. Visualise our results with Lightdash



FORMULA 1

Rough Overview of the Data

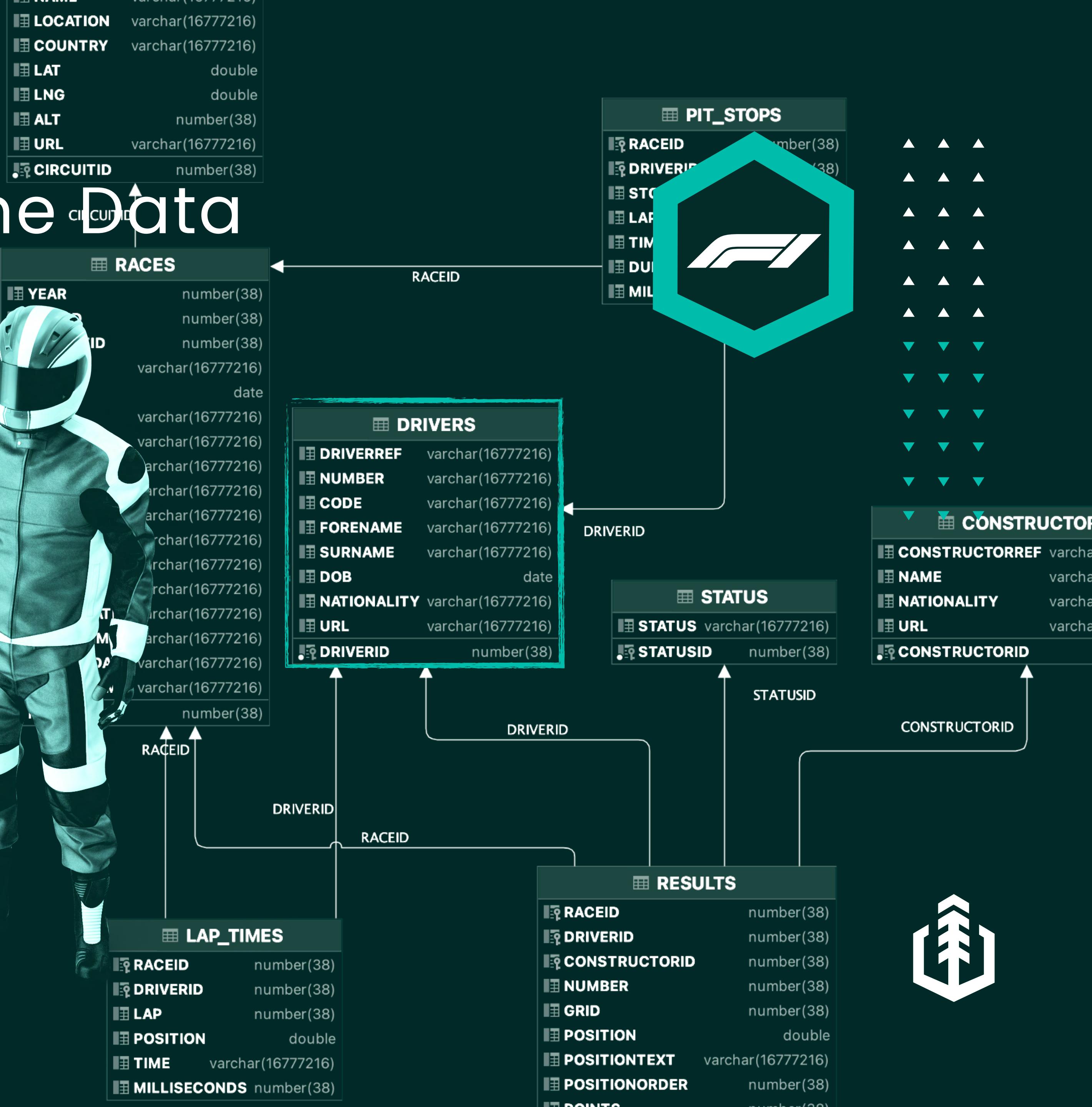
- dimensions: drivers, circuits, constructors, races
- facts: lap_times, pit_stops, results



FORMULA 1

Rough Overview of the Data

- dimensions: drivers, circuits, constructors, races
- facts: lap_times, pit_stops, results



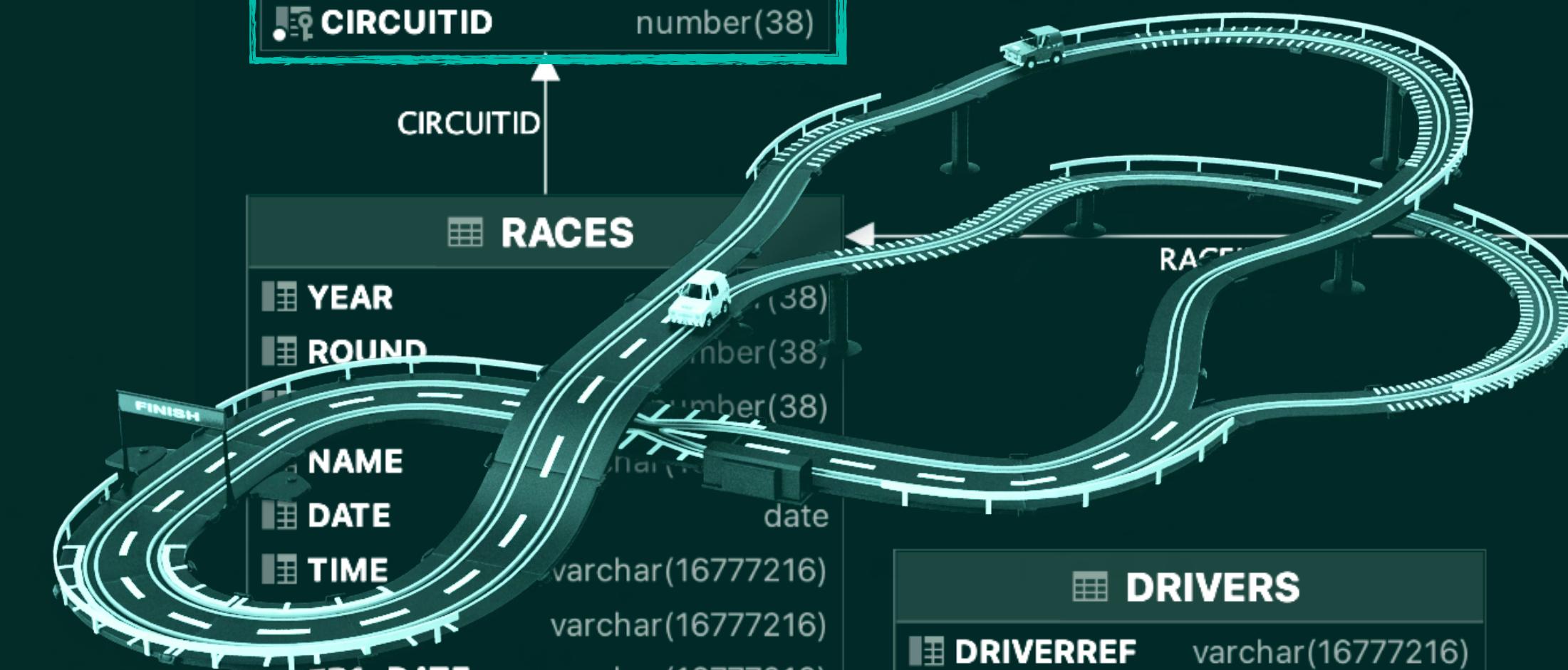
FORMULA 1

Rough Overview of the Data

- dimensions: drivers, circuits, constructors, races
- facts: lap_times, pit_stops, results

CIRCUITS	
CIRCUITREF	varchar(16777216)
NAME	varchar(16777216)
LOCATION	varchar(16777216)
COUNTRY	varchar(16777216)
LAT	double
LNG	double
ALT	number(38)
URL	varchar(16777216)
CIRCUITID	number(38)

RACES	
YEAR	number(38)
ROUND	number(38)
NAME	number(38)
DATE	date
TIME	varchar(16777216)



DRIVERS	
DRIVERREF	varchar(16777216)
NUMBER	varchar(16777216)
CODE	varchar(16777216)
FORENAME	varchar(16777216)
SURNAME	varchar(16777216)
DOB	date
NATIONALITY	varchar(16777216)

STATUS	



PIT_STOPS	
RACEID	number(38)
DRIVERID	number(38)
STOP	number(38)
LAP	number(38)
TIME	varchar(16777216)
DURATION	varchar(16777216)
MILLISECONDS	number(38)

YEAR	number(38)
ROUND	number(38)
CIRCUITID	number(38)

RACEID

MILLISECONDS	number(38)
--------------	------------

FORMULA 1

Rough Overview of the Data

- dimensions: drivers, circuits, constructors, races
- facts: lap_times, pit_tops, results

DRIVERID

STATUS	
STATUS	varchar(16777216)
STATUSID	number(38)

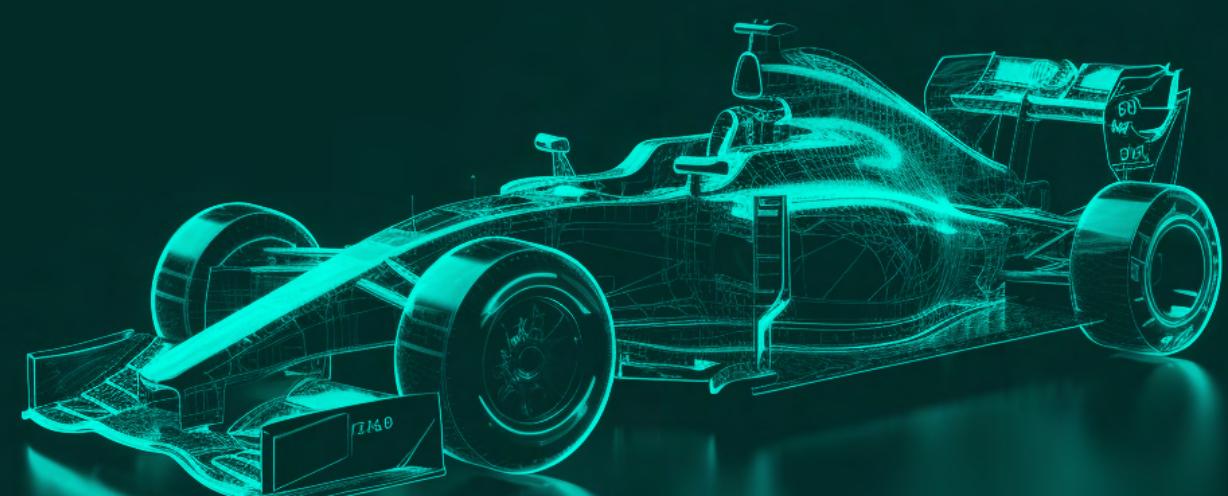
RESULTS

CEID	number(38)
DRIVERID	number(38)
CONSTRUCTORID	number(38)
NUMBER	number(38)
ID	number(38)
SPOSITION	double
SPOSITIONTEXT	varchar(16777216)
SPOSITIONORDER	number(38)
INTS	number(38)
PS	number(38)
TIME	varchar(16777216)
MILLISECONDS	number(38)

CONSTRUCTORS	
CONSTRUCTORREF	varchar(16777216)
NAME	varchar(16777216)
NATIONALITY	varchar(16777216)
URL	varchar(16777216)
CONSTRUCTORID	number(38)

STATUSID

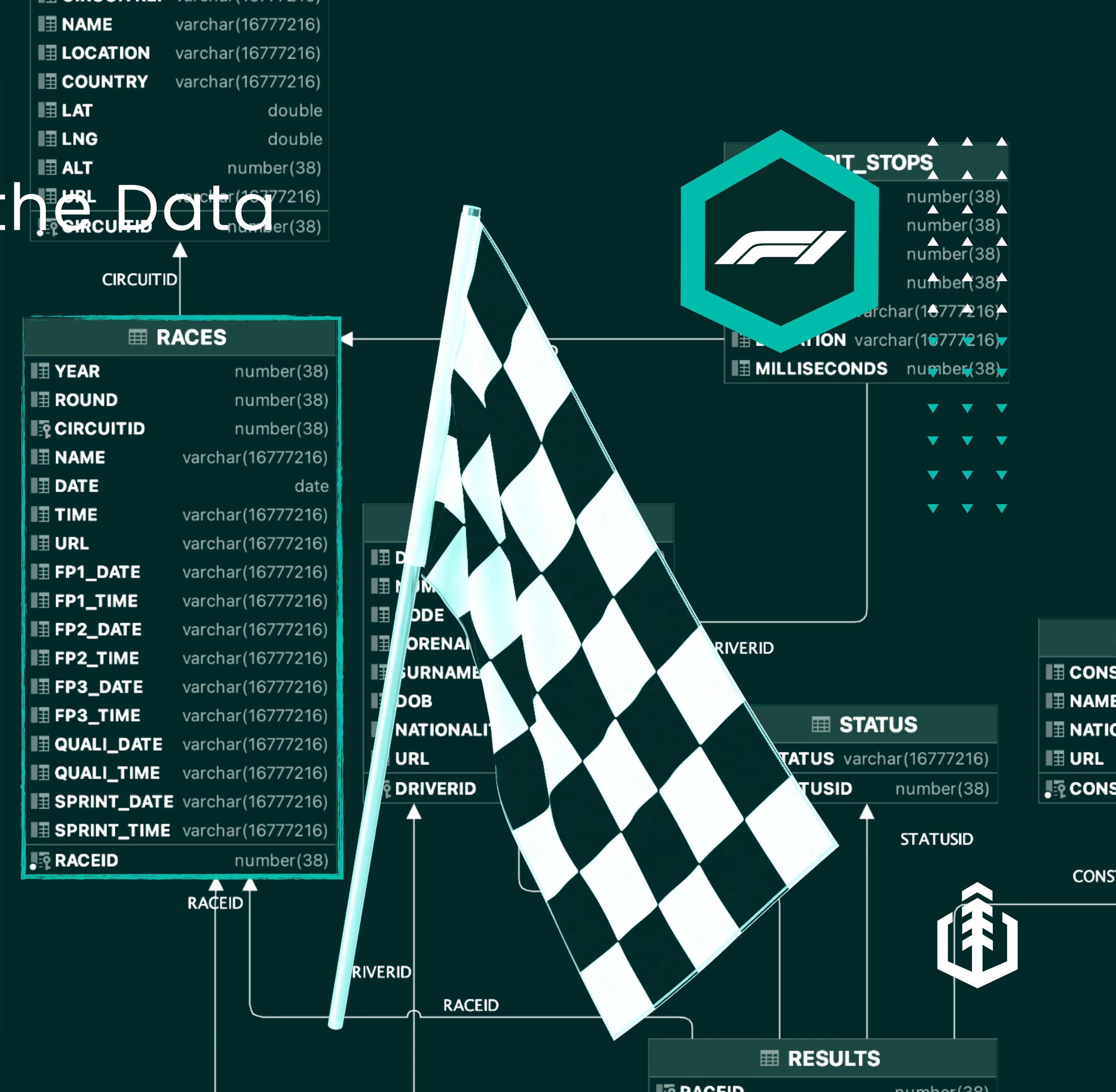
CONSTRUCTORID



FORMULA 1

Rough Overview of the Data

- dimensions: drivers, circuits, constructors, races
- facts: lap_times, pit_stops, results



FORMULA 1

Rough Overview of the Data

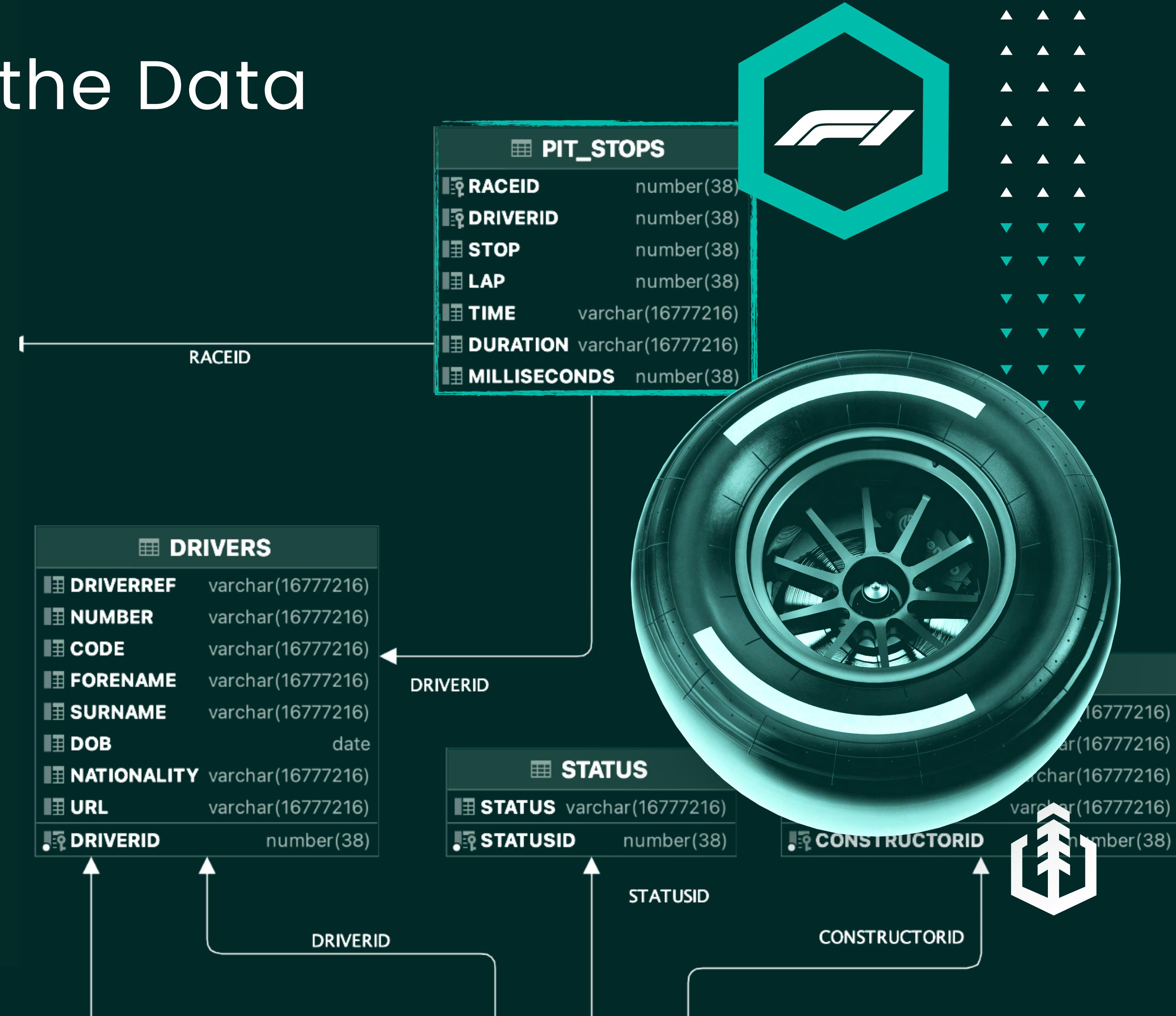
- dimensions: drivers, circuits, constructors, races
- facts: lap_times, pit_tops, results



FORMULA 1

Rough Overview of the Data

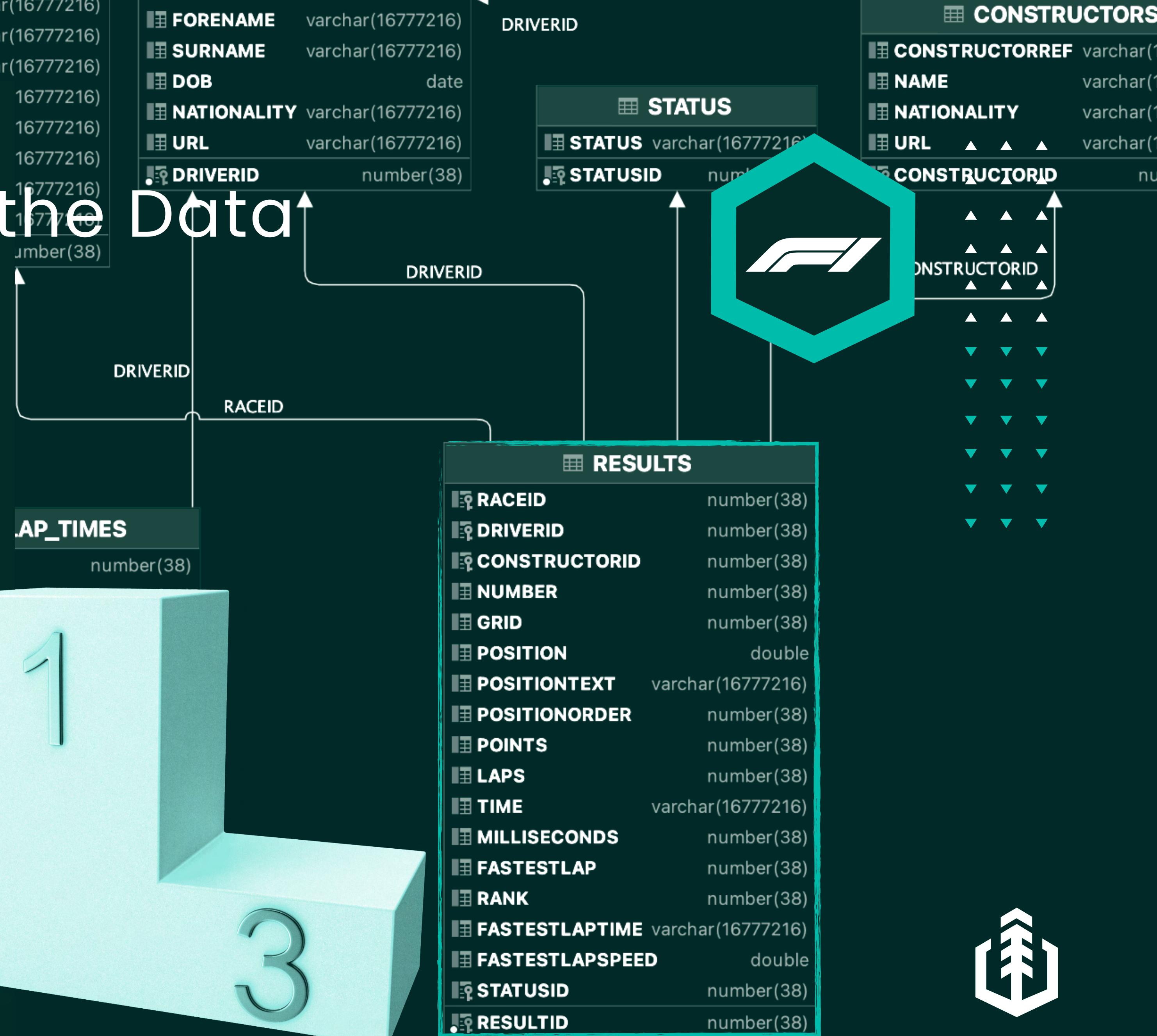
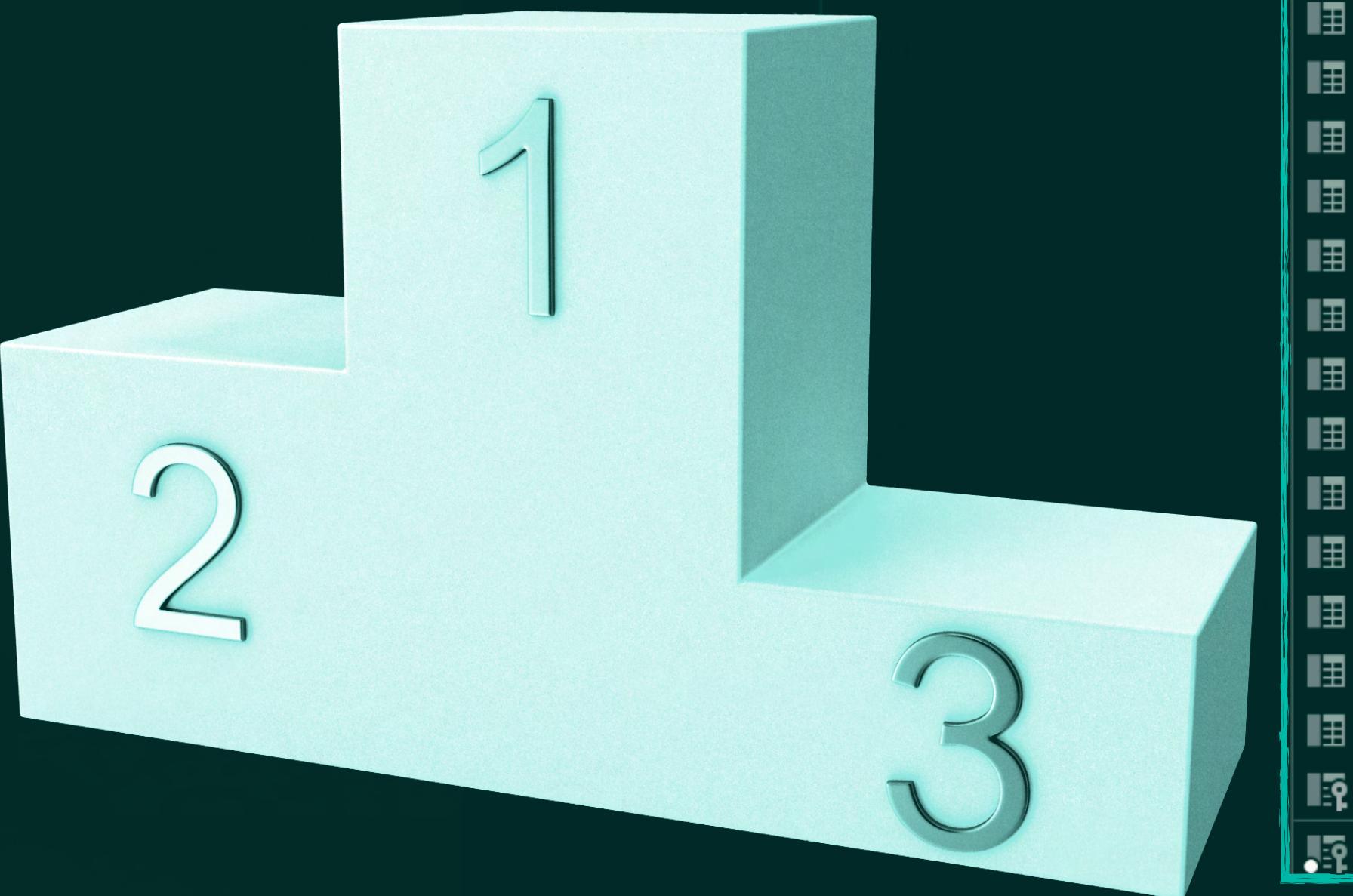
- dimensions: drivers, circuits, constructors, races
- facts: lap_times, pit_stops, results



FORMULA 1

Rough Overview of the Data

- dimensions: drivers, circuits, constructors, races
- facts: lap_times, pit_tops, results



SNOWFLAKE

Snowflake Setup



- ▶ activate the Anaconda Packages (Preview Feature)
- ▶ create a warehouse to be used for dbt
- ▶ that's it already!

AIRBYTE

Ingesting the Data with Airbyte



Connections

NAME	SOURCE NAME	DESTINATION NAME	FREQUENCY	LAST SYNC	ENABLED
File <> Snowflake	File - File	Snowflake - Snowflake	Manual	21 hours ago	<button>Launch</button> <button>⚙️</button>

+ New connection

PYTHON + SNOWPARK

Setup Python/Snowpark for dbt



- just install along dbt also
 - **dbt-snowflake**
 - **snowflake-snowpark-python**
 - **snowflake-connector-python**
- and configure Snowflake in your `profiles.yml`

PYTHON + SNOWPARK

Training a Python Model in dbt/Snowpark



```
● ● ●  
def model(dbt, session):  
    dbt.config(  
        packages=["numpy", "scikit-learn", "pandas"],  
        materialized="table",  
        tags="train",  
    )  
    session.sql("create or replace stage MODELSTAGE").collect()  
    test_train_df = dbt.ref("train_test_dataset")  
    test_train_pd_df = test_train_df.to_pandas()  
    X_train, X_test, y_train, y_test = train_test_split(  
        test_train_pd_df.drop(["target"], axis=1), test_train_pd_df[target_col],  
        train_size=0.7, random_state=42  
    )  
    model = LogisticRegression()  
    model.fit(X_train, y_train)  
    save_file(session, model, "@MODELSTAGE/", "model_v1.joblib")
```

Predicting with a trained Model in dbt/Snowpark



```
● ● ●  
  
def model(dbt, session):  
    dbt.config(  
        packages=[ "snowflake-snowpark-python", "scikit-learn", "pandas" ],  
        materialized="table",  
        tags="predict",  
    )  
    model = load_model_from_stage(session)  
  
    @udf  
    def predict(p_df: T.PandasDataFrame[int, int]) -> T.PandasSeries[int]:  
        pred_array = model.predict(p_df)  
        return pd.Series(pred_array)  
  
    snow_df = dbt.ref("hold_out_dataset_for_prediction").select(*FEATURE_COLS)  
  
    return snow_df.withColumn("predicted", predict(*FEATURE_COLS))
```

Visualisation with Lightdash



Lightdash pulls dimensions and metrics from dbt.

What are the avg lap times split by year and grand prix?

METRIC

DIMENSIONS

Lightdash

- runs ad-hoc queries
- creates charts from queries
- creates dashboards from charts
- organises everything in spaces

LIGHTDASH

Main View of Lightdash

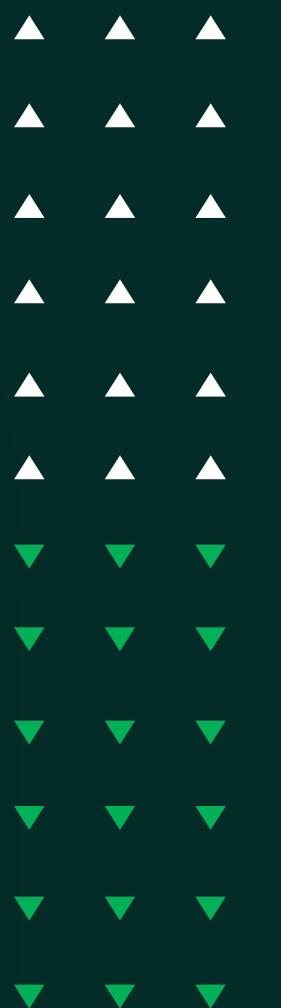


The screenshot shows the Lightdash application interface. At the top, there is a navigation bar with icons for New, Browse, Search, and Help. On the left, a sidebar provides quick access to 'Query from tables', 'Query using SQL runner', 'Dashboard', and 'Space'. The main area features a 'Welcome, Florian!' message and a button to 'Run a query'. Below this is a 'Spaces' section with a box for 'Florian's Space'. The bottom half of the screen displays a list of dashboards, sorted by 'Most popular'. The listed dashboards are:

Name	Last Edited
Race Analysis Dashboard Dashboard • 8 views	an hour ago by Florian Wilhelm
Lap Times over Years Horizontal bar chart • 3 views	an hour ago by Florian Wilhelm

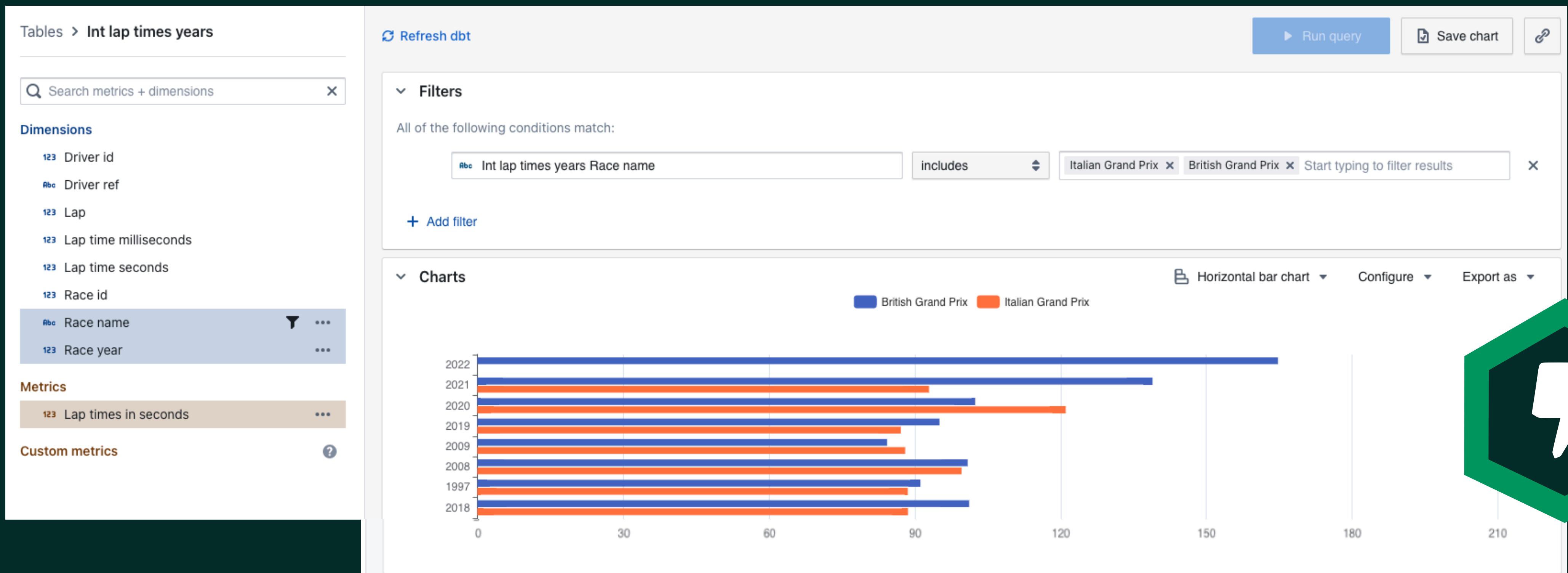


inovex



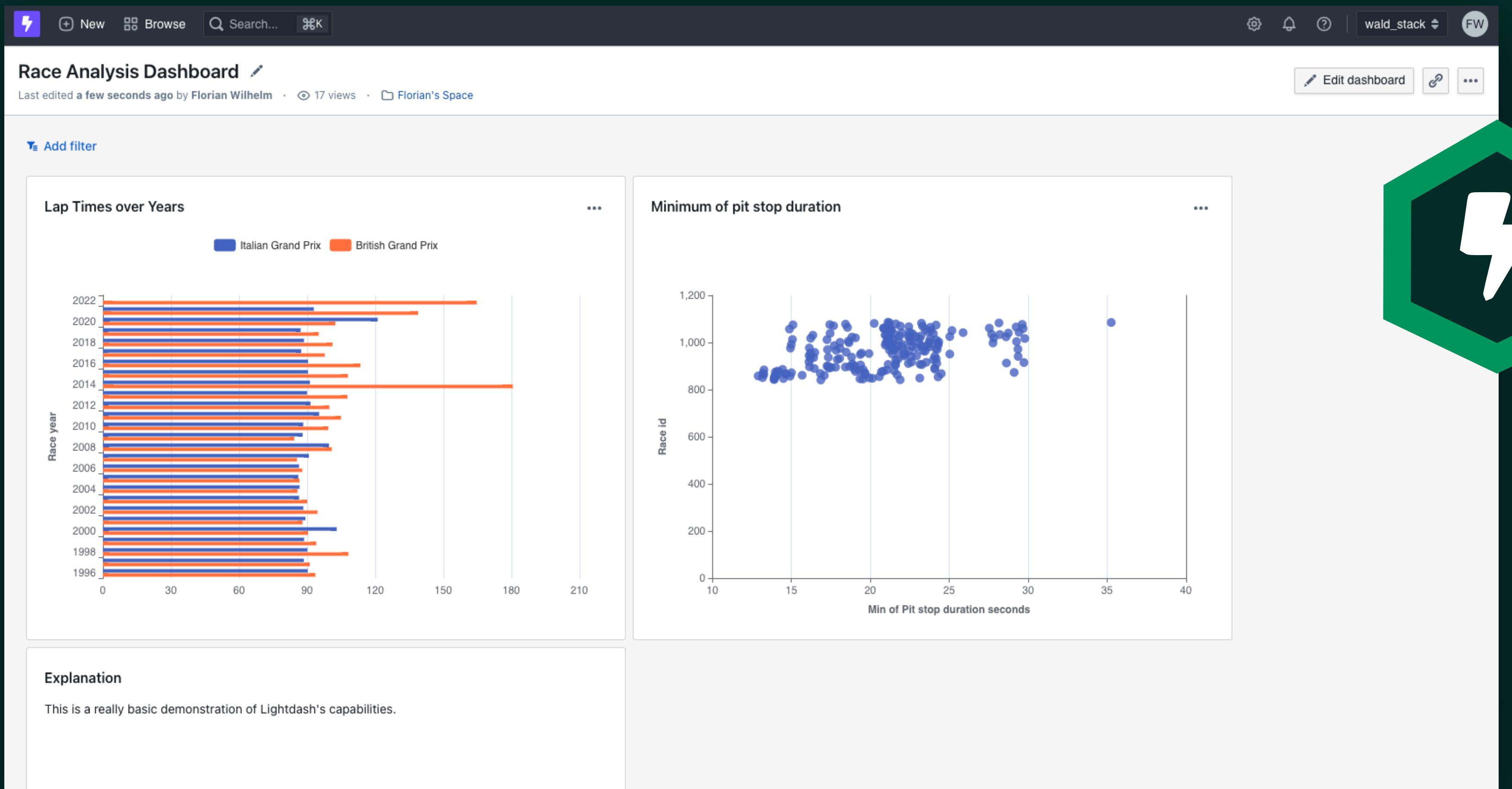
LIGHTDASH

Analysing the Average Lap Times



LIGHTDASH

Example of a Simple Dashboard



FORMULA 1

Analysis and Position Prediction



One more thing...



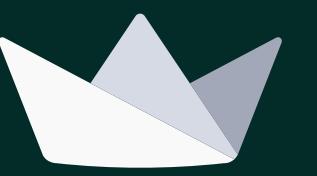
ON MORE THING ON...

Streamlit

- ▶ provides faster way to build and share data apps
- ▶ was recently acquired by Snowflake
- ▶ might be a nice addition to Lightdash for interactive data apps



 snowflake

 Streamlit



WALD STACK

Conclusion

The WALD stack combines 4 powerful technologies that work very well together.

The WALD stack is flexible enough for many use-cases from analytics to building a full-fledged end-to-end data product at scale.



CHEERS TO THE COMMUNITY

Credits & Resources

- ▶ [python-snowpark-formula](#)
[Github](#) repo by [Hope Watson](#)
from dbt labs
- ▶ kind support by [Michael Gorkow](#)
& [Marko Cavar](#) from Snowflake
- ▶ these awesome slides by
[Michael Hofmann](#) from inovex



PyConDE / PyData 2023

Thank you!



Dr. Florian Wilhelm
Head of Data Science

- ✉ florian.wilhelm@inovex.de
- 🌐 inovex.de
- 📷 @inovexlife
- 🐦 @inovexgmbh



waldstack.org

