

The Art and Science of Transportation Research in the AI Era

# A Gentle Introduction to SQL

M.Sc. Hiba Karam



# Learning Goals



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

**#1** Understand what is a database  
and most used types

**#3** How to retrieve data from  
a SQL database

**#2** Understand what is a relational  
database

**#4** Further write Basic SQL syntax

# Lecture Structure



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

- #1** About Database
- #2** Types of Databases
- #3** SQL 101



# #1 About Database



- A database is an **organized collection of structured information, or data, typically stored electronically in a computer system**. Databases are managed using database management systems (DBMS).
- The sum total of the database, the DBMS and the associated applications can be referred to as a **database system**.

## Importance of Database:

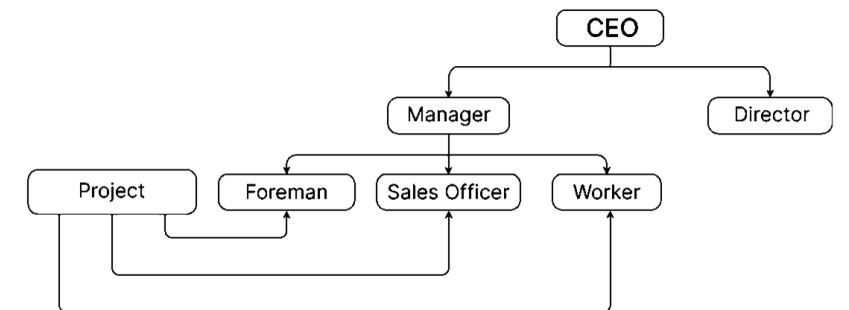
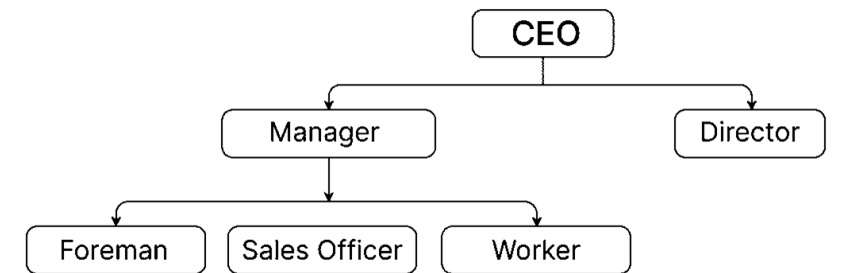
- It gives us a **highly efficient method** for **handling large amount** of different types of data with ease.
- It allows large amount of data to be **stored systematically** and these data to be easily retrieved, filtered, sorted and updated efficiently and accurately.



## #2 Types of Database



- **Object-Oriented Databases:** Represent data as objects these objects can contain both data (attributes) and methods (functions/actions) and can model complex data structures. They are primarily used in specialized applications such as computer-aided design, multimedia.
- **Hierarchical Databases:** Organize data in a tree-like structure with a single root, branches, and leaves. In a hierarchical database, each record has a parent-child relationship making it useful for one-to-many relationships but limited in flexibility.
- **Network Databases:** Similar to hierarchical databases but allow multiple parent nodes, enabling more complex relationships and many-to-many connections.



## #2.1 Relational Database



- **Relational Databases:** organize data into tables with **rows and columns**. They are widely used for structured data and ensure data integrity through relationships between tables.
- Within relational databases, tables are organized in **schemas**: container or namespace within a database that organizes and groups related objects, such as tables, views, indexes, stored procedures, and other database objects. It serves as a way to **logically separate and manage these objects within a database**.

**SQL (Structured Query Language) is a programming language designed for managing data in a relational database.**

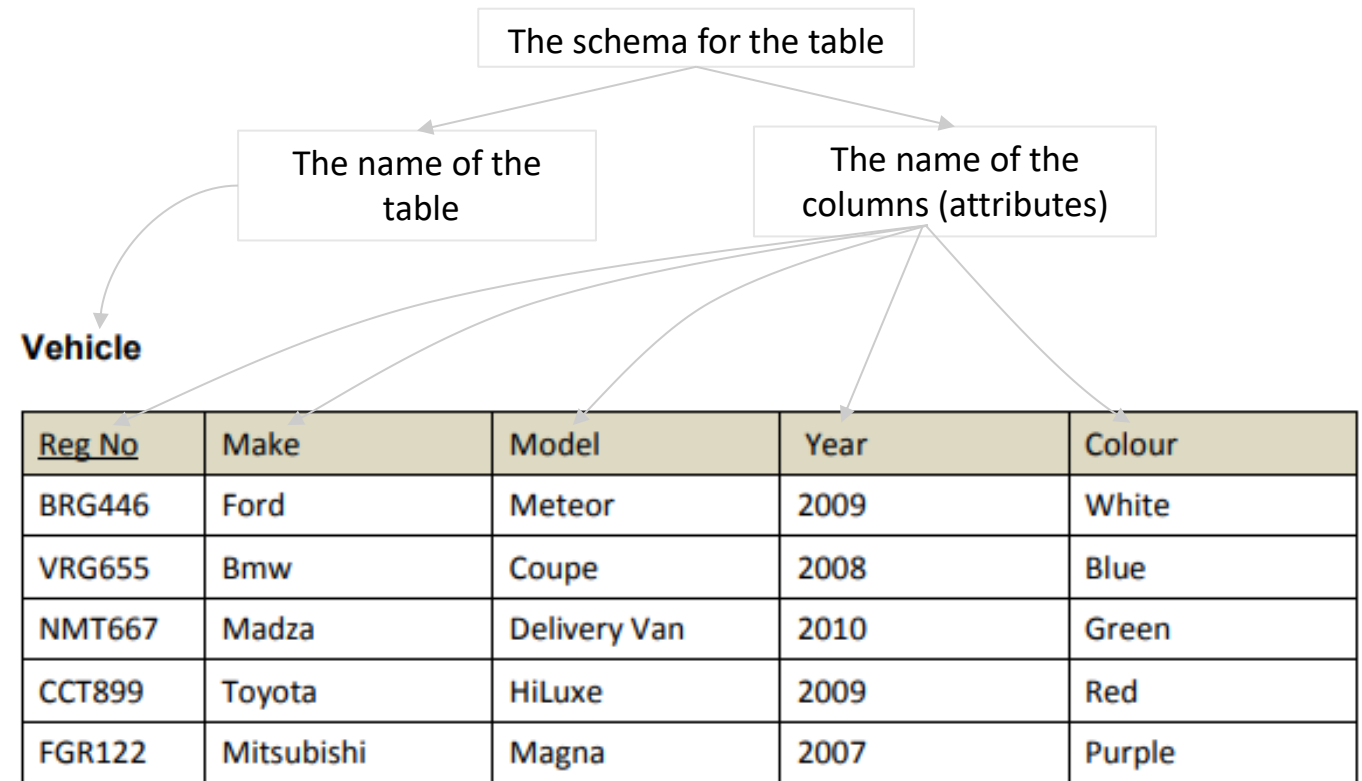
- It was invented in 1974 by IBM and is the most common method of accessing data in relational databases today.



## #2.1 Relational Database



- For every column of every table the schema specifies **allowable value**. For example, **Year** column a is integer... etc
- The set of allowable values for a column is called **domain** or **type of the column**.
- It looks simple, but it is a **very crucial** because RDB are about **structure**. Each car will for sure have the **exact attributes**. This is exactly what we mean by a schema.
- The rest of the table is called **rows**.
- The schema stays the same but the rows change as we add more data.**



Imagine that this table (or relation) has been defined to keep track of vehicles in a company

# #2.1 Relational Database

Table: Artists

ArtistID	Name	BirthYear	Nationality
1	Vincent van Gogh	1853	Dutch
2	Pablo Picasso	1881	Spanish
3	Frida Kahlo	1907	Mexican

ArtistID is the primary key because it uniquely identifies each artist.

Table: Exhibitions

ExhibitionID	ExhibitionName	Location	ArtworkID
1001	Impressionist Masters	The Louvre, Paris	101
1002	War and Peace in Art	Reina Sofía, Madrid	102
1003	Surrealism and Beyond	MoMA, New York	103

ExhibitionID is the primary key because it uniquely identifies each exhibition.

ArtworkID is the primary key because it uniquely identifies each artwork.

Table: Artworks

ArtworkID	Title	YearCreated	ArtistID
101	Starry Night	1889	1
102	Guernica	1937	2
103	The Two Fridas	1939	3

A primary key is a unique identifier for each record in a database table.



## #2.1 Relational Database



Table: Artists

ArtistID	Name	BirthYear	Nationality
1	Vincent van Gogh	1853	Dutch
2	Pablo Picasso	1881	Spanish
3	Frida Kahlo	1907	Mexican

Foreign Key

The tables are linked together using a foreign key (in table Artworks) referring to the primary key (in table Artists).

Table: Artworks

ArtworkID	Title	YearCreated	ArtistID
101	Starry Night	1889	1
102	Guernica	1937	2
103	The Two Fridas	1939	3

The tables are linked together using a foreign key (in table Exhibitions) referring to a primary key (in table Artworks).

Foreign Key

Table: Exhibitions

ExhibitionID	ExhibitionName	Location	ArtworkID
1001	Impressionist Masters	The Louvre, Paris	101
1002	War and Peace in Art	Reina Sofía, Madrid	102
1003	Surrealism and Beyond	MoMA, New York	103

**A foreign key is used to create a relationship between two tables.**

It is a column (or set of columns) in one table that refers to the primary key in another table.

This establishes a link between the records in the two tables.

The action of creating and joining the tables...etc is done using SQL!

## #2.1 Relational Database



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

### Activity

**Goal:** Understand the concept of a database table by creating a table yourself.



Pick a topic or activity you are interested in, for example:

Sports – teams and players

School – students and classes

Others

**Create 2-3 tables to represent the topic or activity.**

**For each table, list the table name and 3-4 attributes.**

**For each table, list the primary and foreign key**



**Table Name:**

**Attributes:**

- Column name (domain)

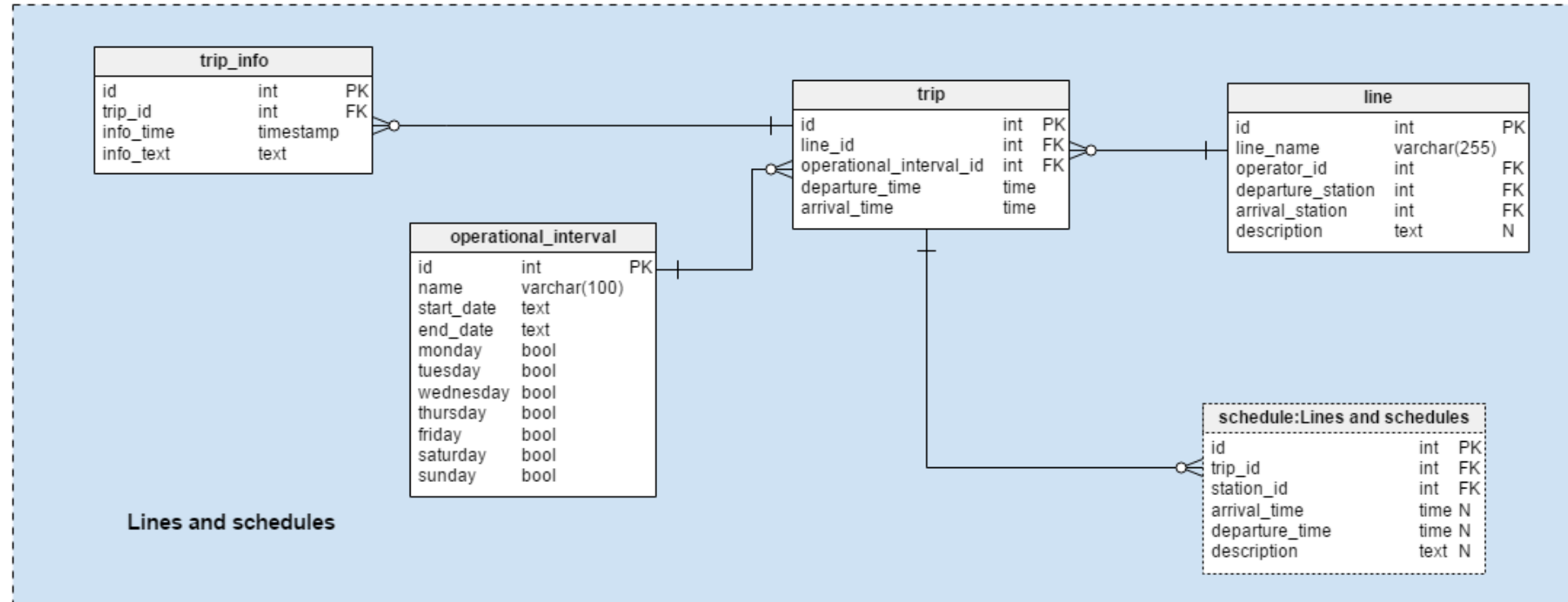
## #2.1 Relational Database



Data modelling is **the process of diagramming data flows.**

It provides a clear and structured visualization of **how data is organized, stored, and related within a database.**

With a clear understanding of how tables are related we can write **more efficient SQL queries.**



Data model in a relational database

We can identify the correct tables to join, select the right columns, and apply filters more accurately, which **improves the performance and accuracy of data retrieval.**

<https://vertabelo.com/blog/traveling-by-bus-or-train-a-transport-hub-database-model/>

## #2.1 Relational Database



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

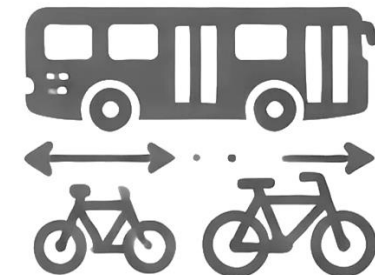


### Data Collection, Management, and Storage

Research often involve large amounts of data from various sources, such as traffic sensors, GPS, surveys, and public transport systems. SQL databases are an **effective way to store, organize, and manage** this data for easy access and analysis.

### Public Transport and Multimodal Studies

SQL can be used to link datasets from various modes of transport (e.g., bus schedules, bike-sharing data, vehicle traffic). Researchers can **run queries to study intermodal relationships**, identifying areas where modes can complement each other or where conflicts arise.

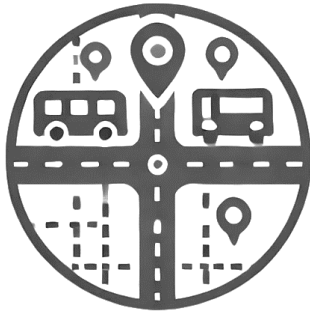




## #2.1 Relational Database



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



### Traffic Pattern Analysis and Modeling

SQL can be used to query and analyze large datasets to **identify traffic patterns**, such as peak traffic periods, congestion points, or accident-prone areas. By applying SQL queries, researchers can derive insights from the data and build traffic models that inform planning decisions.

### Geospatial Analysis with SQL and GIS

SQL can be integrated with Geographic Information Systems (GIS) to **enhance spatial data analysis**. SQL enables researchers to perform spatial queries on transportation networks. For instance, they can query for traffic data within a specific distance from major intersections or analyze traffic flow along specific corridors. Researchers can conduct studies on how traffic flows are influenced by land use patterns or how accessibility to public transport differs across urban areas.



## #2.1 Relational Database



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

There are many relational databases that use SQL (Structured Query Language) as their primary language for interacting with the database, such as PostgreSQL, MySQL, SQLite, and SQL Server. All share the same basic structure of standard SQL, and the key commands are generally similar. However, **there are syntax differences** that are specific to their corresponding RDBMS.

PostgreSQL



## #2.1 Relational Database



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



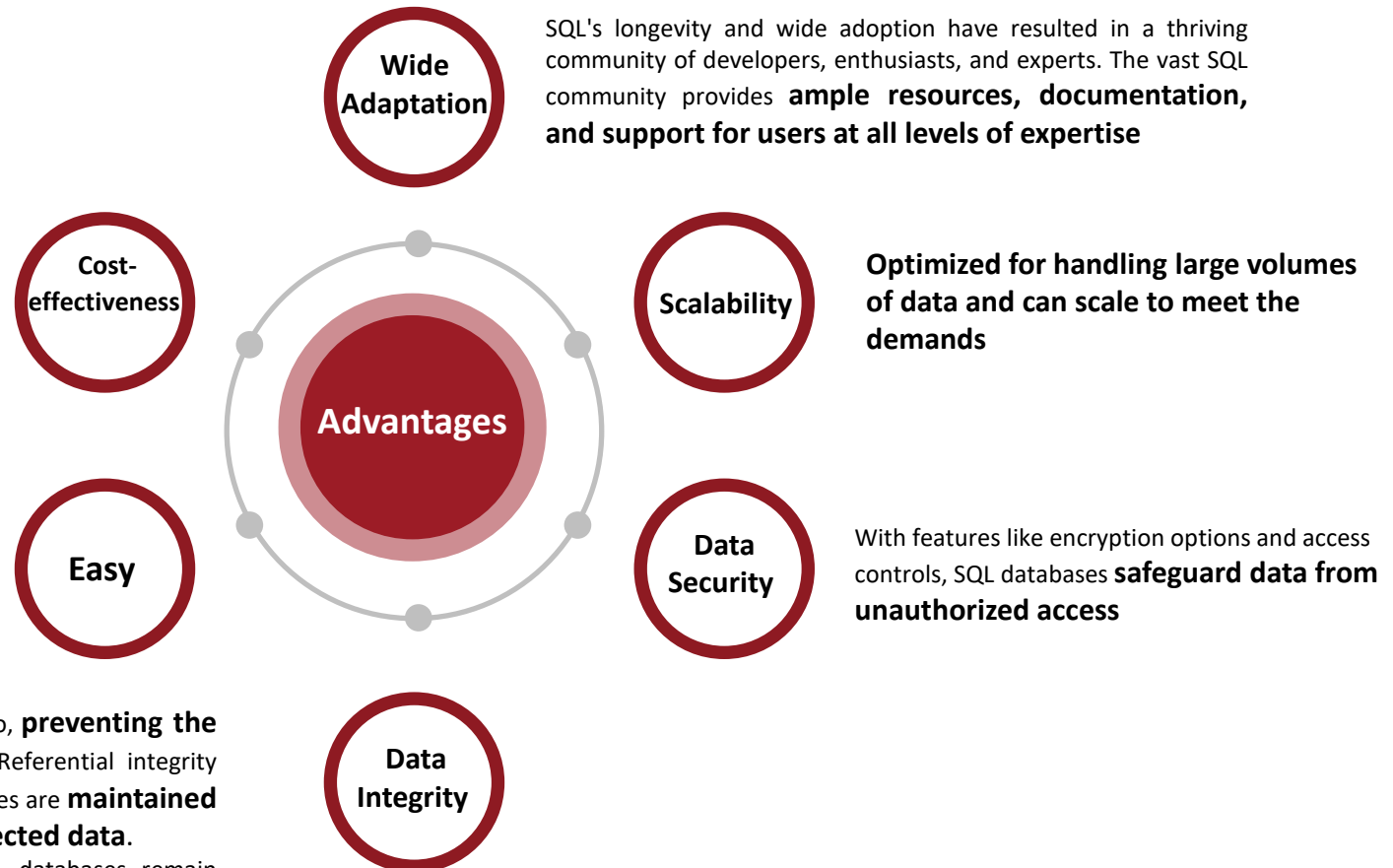
Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

SQL databases, especially open-source options like MySQL and PostgreSQL, offer a **cost-effective alternative** to proprietary database systems

**Easy data retrieval and manipulation**

Constraints define rules that data must adhere to, **preventing the entry of invalid or inconsistent data**. Referential integrity ensures that relationships between different tables are **maintained correctly, avoiding orphaned or disconnected data**.

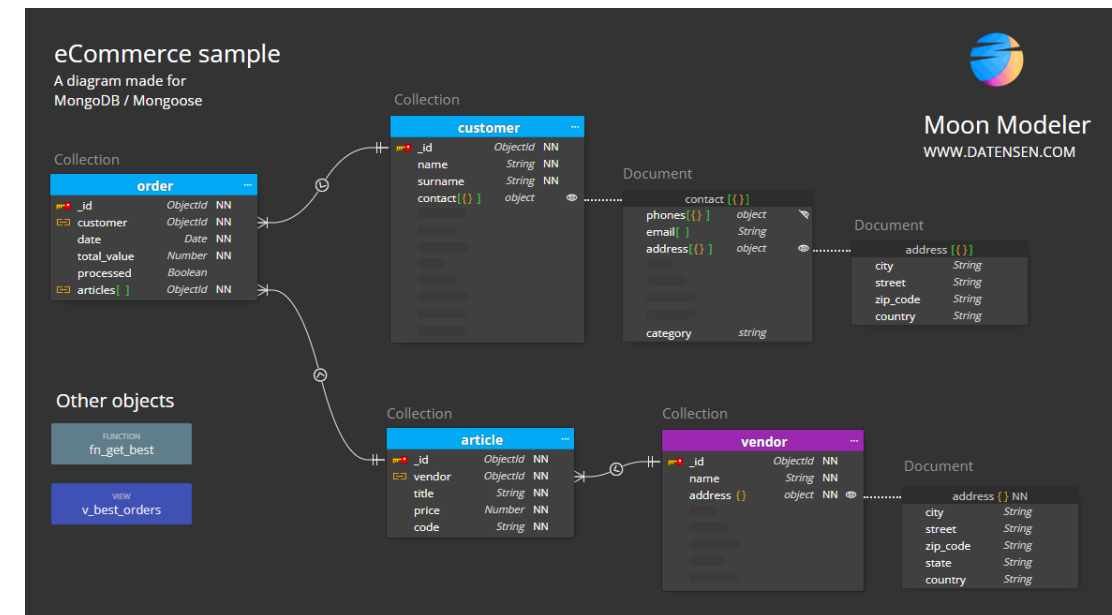
By adhering to these essential principles, SQL databases remain accurate, reliable, and consistent



## #2.2 NoSQL



- NoSQL, also referred to as “not only SQL” or “non-SQL refers to Non-Relational Databases
- is commonly referenced in relation to SQL.
- NoSQL databases, in contrast to conventional relational databases, **do not rely on a predefined schema** and house data within one data structure, such as JSON document. Popular database of NoSQL are MongoDB and Cassandra.
- Offer **greater flexibility** in processing unstructured or semi-structured data.
- NoSQL Non-relational databases have existed since the late 1960s, but the name "NoSQL" was only coined in the early 2000s.



Non-Relational Database

**Note: we want you to be familiar with the term NoSQL**



## #2.3 SQL vs NoSQL



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



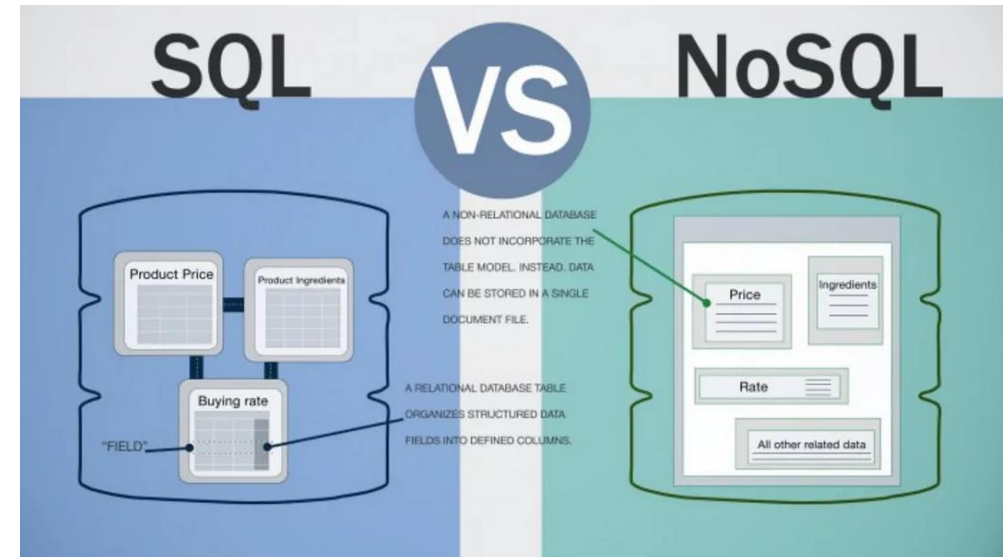
Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

**Structured predefined schema:** SQL databases use a **predefined and fixed schema**, which dictates how data is organized. Data is stored in **tables, with rows and columns**. Each table is related to another table through **foreign keys**.

**Example:** In a transportation database, you might have a Vehicles table with columns like VehicleID, Make, Model, and Year. **Each row represents a different vehicle and will have the same exact columns.**

**Flexible Schema:** NoSQL databases **do not require a fixed schema**. They are designed to handle **unstructured or semi-structured data** and can store data in formats like documents, key-value pairs, graphs, or wide columns.

**Example:** In a document-based NoSQL database like MongoDB, a Vehicle might be stored as a document (similar to a JSON object) with fields like VehicleID, Make, Model, and Year. **Different vehicles can have different fields without adhering to a strict schema.**



# #3 SQL 101



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

- SQL is needed by anyone who needs to create, modify, or communicate with relational databases.
- Commands such as SELECT, UPDATE, INSERT, DELETE, and so on remain largely unchanged.

```
SELECT * FROM employees;
```

This query selects all columns from the "employees" table.

```
UPDATE employees SET salary = 50000 WHERE id = 1;
```

This updates the salary to 50,000 for the employee with ID 1.

```
INSERT INTO employees (name, salary) VALUES ('John Doe', 45000);
```

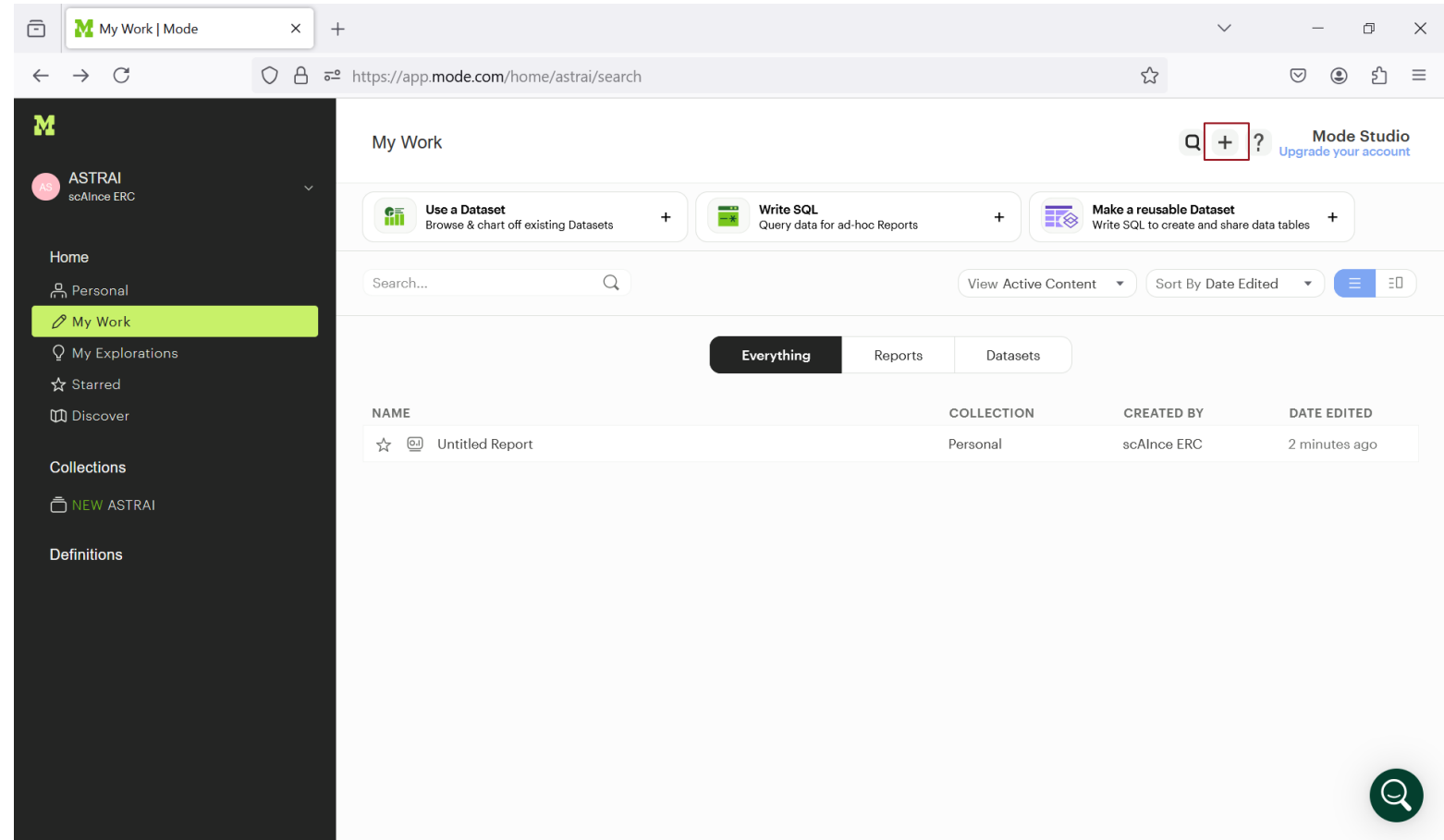
This inserts a new employee named John Doe with a salary of 45,000.

```
DELETE FROM employees WHERE id = 1;
```

This deletes the employee with ID 1 from the "employees" table.

# #3 SQL 101

- Mode is an open data analysis platform (also have paid version) that allows users to query data using SQL and also provides learning materials and a space for practice.
- Mode's SQL editor is designed to support standard SQL commands however, there may be minor syntax variations and specific functions unique to each database system.
- SQL commands are not case sensitive, but the data it self is!!



Mode Interface

# #3.1 SELECT FROM



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



## Basic syntax: SELECT and FROM

- There are two required ingredients in any SQL query: **SELECT** and **FROM**—and they have to be in that order.
- SELECT indicates which columns you'd like to view, and FROM identifies the table that they live in.
- \* means get me all the columns in the specific table.**

The screenshot shows the ASTRAI web interface. At the top, there's a header with the user 'scAlnce ERC' and the application name 'ASTRAI'. Below the header, on the left, is a sidebar with options like 'Report Builder' and 'Add Notebook'. The main area displays a SQL query editor with the following code:

```
1  
2  
3 SELECT *  
4 FROM tutorial.flights  
5  
6 SELECT *  
7 FROM tutorial.flights_revenue
```

Below the query editor, there's a 'Run Selected' button and a 'Limit 100' checkbox. The results are displayed in a table with the following columns: actual\_arrival\_time, actual\_flight\_time, acutal\_departure\_time, air\_time, and air\_traffic\_delay. The table shows 9 rows of data. On the right side, there's a 'Mode Public Warehouse (everyone)' dropdown and a search bar. Below the search bar, there's a list of tables including 'tutorial.nominee\_information', 'animal\_crossing\_villagers', 'animal\_crossing\_wall\_mounted', 'animal\_crossing\_wallpaper', 'animal\_crossing\_accessories', 'billboard\_top\_100\_year\_end', 'city\_populations', 'crunchbase\_acquisitions', 'crunchbase\_acquisitions\_clean\_date', 'crunchbase\_companies', and 'crunchbase\_companies\_clean\_date'. The 'flights' table is selected, and its schema is shown on the right, listing columns like 'actual\_arrival\_time', 'actual\_flight\_time', 'acutal\_departure\_time', 'air\_time', 'air\_traffic\_delay', 'airline\_code', 'airline\_name', 'arrival\_delay', and 'cancellation\_reason'.

	actual_arrival_time	actual_flight_time	acutal_departure_time	air_time	air_traffic_delay
1	1339	224	655	200	
2	951	115	656	97	
3	1429	180	1129	152	
4	944	140	724	108	
5	1300	146	1034	132	
6	848	118	550	92	
7	1045	161	804	140	
8	1246	111	1055	89	
9	755	66	549	41	



# #3.1 SELECT FROM

M

AS

scAlnce ERC > ASTRAI

Report Share View

ASTRAI

Report Builder

Add Notebook

DATA

Query 1

New chart

Query 1

Run Selected Limit 100 Format View history

```
1
2
3 SELECT *
4 FROM tutorial.flights
5
6 SELECT *
7 FROM tutorial.flight_revenue
```

Succeeded

	destination_airport	cargo_rev	first_class_rev	business_class_rev	coach_rev	id
1	SFO	10239	15747	12119	11782	
2	LAX	17437	18874	10931	23363	
3	JFK	10272	19153	10396	12549	
4	ANC	10099	16796	6568	25099	
5	LHR	13658	16068	13497	23195	
6	ORD	14490	15997	8916	11690	
7	DEN	16940	13324	12968	29265	
8	DFW	13543	13976	8381	25007	
9	ABQ	14179	19775	5010	11110	

Page 1 of 1 Showing rows 1-14 Columns Size Run a few seconds Executed in

Mode Public Warehouse (everyone)

Search this Connection...

tutorial.nominee\_information

animal\_crossing\_villagers

animal\_crossing\_wall\_mounted

animal\_crossing\_wallpaper

animial\_crossing\_accessories

billboard\_top\_100\_year\_end

city\_populations

crunchbase\_acquisitions

crunchbase\_acquisitions\_clean\_date

crunchbase\_companies

crunchbase\_companies\_clean\_date

crunchbase\_investments

crunchbase\_investments\_part1

crunchbase\_investments\_part2

dc\_bikeshare\_q1\_2012

dunder\_mifflin\_paper\_sales

excel\_sql\_inventory\_data

excel\_sql\_transaction\_data

flight\_revenue

flights

global\_weekly\_charts\_2013\_2014

housing\_units\_completed\_us

# #3.1 SELECT FROM

You can also select a specific column/s

ASTRAI

Report Builder

Add Notebook

DATA

Query 1

New chart

Query 1

Do not forget ,

Run Selected

Limit 100

Format

View history

```
1
2
3 SELECT origin_airport,
4       destination_airport,
5       scheduled_departure_time,
6       scheduled_arrival_time
7 FROM tutorial.flights
8
9 SELECT *
10 FROM tutorial.flight_revenue
```

Succeeded

Data

Fields

Source

Export

Copy

	origin_airport	destination_airport	scheduled_departure_time	scheduled_arrival_time
1	SNA	ATL	645	1356
2	AUS	ATL	700	1008
3	JFK	FLL	1133	1449
4	ATL	BDL	727	941
5	BDL	ATL	1039	1325
6	ICT	ATL	557	905
7	LGA	PBI	805	1106
8	DCA	ATL	1100	1255
9	VPS	ATL	600	804

Page 1 of 1 Showing rows 1-100 Columns Size Run a few seconds Executed in

Mode Public Warehouse (everyone)

Search this Connection...

tutorial.nominee\_information

animal\_crossing\_villagers

animal\_crossing\_wall\_mounted

animal\_crossing\_wallpaper

animal\_crossing\_accessories

billboard\_top\_100\_year\_end

city\_populations

crunchbase\_acquisitions

crunchbase\_acquisitions\_clean\_date

crunchbase\_companies

crunchbase\_companies\_clean\_date

crunchbase\_investments

crunchbase\_investments\_part1

crunchbase\_investments\_part2

dc\_bikeshare\_q1\_2012

dunder\_mifflin\_paper\_sales

excel\_sql\_inventory\_data

excel\_sql\_transaction\_data

flight\_revenue

flights

global\_weekly\_charts\_2013\_2014

housing\_units\_completed\_us

# #3.1 SELECT FROM



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

Writing a new Query

## #3.1 SELECT FROM



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt



1. Get Familiar with the columns in both tables `tutorial.flights` and `tutorial.flight_revenue` or any other tables using (\*).
2. Select any four or more columns.



## #3.2 WHERE Filter



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

- The **WHERE** clause is used to filter records.
- It is used to extract only those records that **fulfill a specified condition**.
- Remember when using SQL, entire rows of data are **preserved together**. Meaning the operations typically affect entire rows of data, rather than individual columns or cells.
- The **clause order is important**. Therefore, writing what when is critical.
- The most basic way to filter data is using **1) comparison operators**.
- The easiest way to understand them is to start by looking at a list of them:

Equal to	=
Not equal to	<> or !=
Greater than	>
Less than	<
Greater than or equal to	>=
Less than or equal to	<=

# #3.2 WHERE Filter



Document

ASTRAI

Report Builder

Add Notebook

DATA

Explore

New chart

Where

New chart

Where

Run Selected

Limit 100

Format

View history

1 SELECT \*

2 FROM tutorial.flights

3 WHERE origin\_airport = 'JFK'

Succeeded

		origin_airport	origin_city	origin_state	scheduled_arrival_time	scheduled_departure_time
1	0	JFK	New York	New York	1449	1133
2	0	JFK	New York	New York	1810	1520
3	0	JFK	New York	New York	1304	815
4	0	JFK	New York	New York	1101	811
5	0	JFK	New York	New York	1032	805
6	0	JFK	New York	New York	1642	1152
7	0	JFK	New York	New York	1210	845
8	0	JFK	New York	New York	1428	1120
9	0	JFK	New York	New York	1219	815

Page 1 of 1 Showing rows 1-100 Columns Size Run a few Executed in

Mode Public Warehouse (everyone)

Search this Connection...

tutorial.nominee\_information

animal\_crossing\_accessories

billboard\_top\_100\_year\_end

city\_populations

crunchbase\_acquisitions

crunchbase\_acquisitions\_clean\_date

crunchbase\_companies

crunchbase\_companies\_clean\_date

crunchbase\_investments

crunchbase\_investments\_part1

crunchbase\_investments\_part2

dc\_bikeshare\_q1\_2012

dunder\_mifflin\_paper\_sales

excel\_sql\_inventory\_data

excel\_sql\_transaction\_data

flight\_revenue

flights

global\_weekly\_charts\_2013\_2014

housing\_units\_completed\_us

kag\_conversion\_data

nominee\_filmography

nominee\_information

## #3.2 WHERE Filter



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

2) **Logical operators** allow you to use multiple comparison operators in one query.

ASTRAI

Report Builder

Add Notebook

DATA

Explore

New chart

Where

New chart

Where

Run Selected

Limit 100

Format

View history

1 SELECT \*

2 FROM tutorial.flights

3 WHERE origin\_airport = 'JFK'

4

5

6 SELECT airline\_name,

7 | | origin\_airport,

8 | | destination\_airport,

9 | | air\_traffic\_delay,

10 | | day\_of\_week

11 FROM tutorial.flights

12 WHERE origin\_airport = 'JFK' AND destination\_city = 'Atlanta'

Succeeded

	airline_name	origin_airport	destination_airport	air_traffic_delay	day_of_week
88	JetBlue Airways	JFK	ATL		Saturday
89	JetBlue Airways	JFK	ATL		Sunday
90	JetBlue Airways	JFK	ATL		Monday
91	JetBlue Airways	JFK	ATL		Tuesday
92	JetBlue Airways	JFK	ATL	23	Wednesday
93	Delta Air Lines Inc.	JFK	ATL		Friday
94	Delta Air Lines Inc.	JFK	ATL		Friday
95	Delta Air Lines Inc.	JFK	ATL	15	Friday
96	Delta Air Lines Inc.	JFK	ATL		Friday

Page 1 of 1 Showing rows 1-100 Columns Size Run a few seconds Executed in

Mode Public Warehouse (everyone)

Search this Connection...

tutorial.nominee\_information

animal\_crossing\_accessories

billboard\_top\_100\_year\_end

city\_populations

crunchbase\_acquisitions

crunchbase\_acquisitions\_clean\_date

crunchbase\_companies

crunchbase\_companies\_clean\_date

crunchbase\_investments

crunchbase\_investments\_part1

crunchbase\_investments\_part2

flights

# arrival\_delay number

T cancellation\_reason string

# carrier\_delay number

day datetime

# departure\_delay number

T destination\_airport string

T destination\_city string

T destination\_state string

# distance number

## #3.2 WHERE Filter



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

Here are more logical operators:

- **LIKE** allows you to match similar values, instead of exact values.
- **IN** allows you to specify a list of values you'd like to include.
- **BETWEEN** allows you to select only rows within a certain range.
- **IS NULL** allows you to select rows that contain no data in a given column.
- **AND** allows you to select only rows that satisfy two conditions.
- **OR** allows you to select rows that satisfy either of two conditions.
- **NOT** allows you to select rows that do not match a certain condition.

What is the difference between  
zero and null value?



## #3.2 WHERE Filter



ASTRAI

Where

Run Selected Limit 100 Format View history

```
1 SELECT *
2   FROM tutorial.flights
3  WHERE origin_airport = 'JFK'
4
5
6
7 SELECT destination_airport,
8        first_class_rev
9   FROM tutorial.flight_revenue
10  WHERE first_class_rev BETWEEN 16000 AND 17000
11
12
13
14
```

Succeeded

Data	Fields	Source
	destination_airport	first_class_rev
1	SFO	15747
2	LAX	18874
3	JFK	19153
4	ANC	16796
5	LHR	16068
6	ORD	15997
7	DEN	13324
8	DFW	13976
9	ABQ	19775

Page 1 of 1 Showing rows 1-14 Columns Size Run a few seconds Executed in

ASTRAI

Where

Run Selected Limit 100 Format View history

```
1 SELECT *
2   FROM tutorial.flights
3  WHERE origin_airport = 'JFK'
4
5
6
7 SELECT destination_airport,
8        first_class_rev
9   FROM tutorial.flight_revenue
10  WHERE first_class_rev BETWEEN 16000 AND 17000
11
12
13
14
```

Succeeded

Data	Fields	Source
	destination_airport	first_class_rev
1	ANC	16796
2	LHR	16068

Page 1 of 1 Showing rows 1-2 Columns Size Run a few seconds Executed in



What is the difference between these two queries?

## #3.3 ORDER BY Sorting



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

Once you've learned how to filter data, it's time to learn how to sort data. The **ORDER BY** clause allows you to **reorder your results** based on the data in one or more columns. The ORDER BY keyword sorts the records in **ascending order by default**. To sort the records in descending order, use the DESC keyword.

The screenshot shows the Mode Studio interface. The top bar includes the Mode logo, user profile 'scAlnce ERC', and the workspace name 'ASTRAI'. The left sidebar shows navigation options like 'Report Builder', 'Add Notebook', and 'DATA' with sub-options 'Explore' and 'Where'. The main area displays a SQL query in a dark-themed editor:

```
SELECT airline_name,
       origin_airport,
       destination_airport,
       air_traffic_delay,
       day_of_week
FROM tutorial.flights
WHERE origin_airport = 'JFK' AND destination_city = 'Atlanta'
ORDER BY day_of_week
```

Below the query editor, a table of results is shown with columns: airline\_name, origin\_airport, destination\_airport, air\_traffic\_delay, and day\_of\_week. The results are sorted by day\_of\_week in ascending order.

	airline_name	origin_airport	destination_airport	air_traffic_delay	day_of_week
1	JetBlue Airways	JFK	ATL	0	Friday
2	Delta Air Lines Inc.	JFK	ATL	0	Friday
3	JetBlue Airways	JFK	ATL	26	Friday
4	Delta Air Lines Inc.	JFK	ATL	0	Friday
5	Delta Air Lines Inc.	JFK	ATL	0	Friday
6	Delta Air Lines Inc.	JFK	ATL	0	Friday
7	JetBlue Airways	JFK	ATL	0	Friday
8	Delta Air Lines Inc.	JFK	ATL	0	Friday
9	Delta Air Lines Inc.	JFK	ATL	0	Friday

On the right side, there is a search bar and a list of connections under 'Mode Public Warehouse (everyone)'. A large red question mark icon is overlaid on the right side of the image.



What changed?



## #3.3 ORDER BY Sorting



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt



3. Pick a flight itinerary based on two criteria.
4. Order By a column of your choosing.

## #3.4 AGGREGATION FUNCTIONS



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

- The functions themselves **are the same ones** you will find in Excel or any other analytics program.
- They all **aggregate across the entire table**.

Here are the aggregation functions:

- **COUNT** counts how many rows are in a particular column.
- **SUM** adds together all the values in a particular column.
- **MIN** and **MAX** return the lowest and highest values in a particular column, respectively.
- **AVG** calculates the average of a group of selected values.

# #3.4 AGGREGATION FUNCTIONS



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

Report Builder

Add Notebook

DATA

AGGR

New chart

Explore

New chart

Where

New chart

Run Selected

Limit 100

Format

View history

```
1 SELECT SUM (first_class_rev) AS total_FC,
2      SUM (cargo_rev) AS total_cargo,
3      SUM (business_class_rev) AS total_BC,
4      SUM (coach_rev) AS total_coach
5 FROM tutorial.flight_revenue
```

Ready

	total_fc	total_cargo	total_bc	total_coach
1	225462	198808	140512	273218

Mode Public Warehouse (everyone)

Search this Connection...

tutorial.flights

animal\_crossing\_accessories

billboard\_top\_100\_year\_end

city\_populations

crunchbase\_acquisitions

crunchbase\_acquisitions\_clean\_date

crunchbase\_companies

crunchbase\_companies\_clean\_date

crunchbase\_investments

crunchbase\_investments\_part1

crunchbase\_investments\_part2

flight\_revenue

destination\_airport string

cargo\_rev number

first\_class\_rev number

business\_class\_rev number

coach\_rev number

id serial

Page 1 of 1

Showing rows 1-1

Columns

Size

Run a few seconds

Executed in

## #3.4 AGGREGATION FUNCTIONS



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt



5. The lowest revenue in `first_class_rev` and highest revenue in `business_class_rev`

# #3.4 AGGREGATION FUNCTIONS

M

AS

scAlnce ERC > ASTRAI

Report ▾ Share View

Q

+

?

Mode Studio

Upgrade your account

ASTRAI

AGGR

Report Builder

Add Notebook

DATA

AGGR

Explore

Where

Run Selected

Limit 100

Format

View history

1

2

3

4

5

6

7

8

9

10

11

12

SELECT SUM (first\_class\_rev) AS total\_FC,

SUM (cargo\_rev) AS total\_cargo,

SUM (business\_class\_rev) AS total\_BC,

SUM (coach\_rev) AS total\_coach

FROM tutorial.flight\_revenue

SELECT MIN (first\_class\_rev) AS FC\_lowest,

MAX (business\_class\_rev) AS BC\_highest

FROM tutorial.flight\_revenue

Succeeded

Data

Fields

Source

1

fc\_lowest

bc\_highest

11804

14455

Page 1 of 1

Showing rows 1-1

Columns

Size

Run a few seconds

Executed in

Mode Public Warehouse (everyone) ▾

Search this Connection...

tutorial.flights

animial\_crossing\_accessories

billboard\_top\_100\_year\_end

city\_populations

crunchbase\_acquisitions

crunchbase\_acquisitions\_clean\_date

crunchbase\_companies

crunchbase\_companies\_clean\_date

crunchbase\_investments

crunchbase\_investments\_part1

crunchbase\_investments\_part2

flight\_revenue

destination\_airport

cargo\_rev

first\_class\_rev

business\_class\_rev

coach\_rev

id

11.12.2024 | ASTRAI | W09 | 35

# #3.4 AGGREGATION FUNCTIONS



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt

M

AS

scAlnce ERC > ASTRAI

Report Share View

Q

+

?

Mode Studio

Upgrade your account

ASTRAI

AGGR

Run Selected

Limit 100

Format

View history

Report Builder

Add Notebook

DATA

AGGR

New chart

Explore

New chart

Where

New chart

```
6
7
8 SELECT MIN (first_class_rev) AS FC_lowest,
9         MAX (business_class_rev) AS BC_highest
10 FROM tutorial.flight_revenue
11
12 SELECT destination_airport, first_class_rev AS FC_lowest
13 FROM tutorial.flight_revenue
14 ORDER BY first_class_rev ASC
15 LIMIT 1
```

Succeeded

Data

Fields

Source

Export

Copy

	destination_airport	fc_lowest
1	WAS	11804

Page 1 of 1 Showing rows 1-1 Columns Size Run a few seconds Executed in

Mode Public Warehouse (everyone)

Search this Connection...

tutorial.flights

animial\_crossing\_accessories

billboard\_top\_100\_year\_end

city\_populations

crunchbase\_acquisitions

crunchbase\_acquisitions\_clean\_date

crunchbase\_companies

crunchbase\_companies\_clean\_date

crunchbase\_investments

crunchbase\_investments\_part1

crunchbase\_investments\_part2

flight\_revenue

destination\_airport string

cargo\_rev number

first\_class\_rev number

business\_class\_rev number

coach\_rev number

id serial





TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



Institut für  
Verkehrsplanung  
und Verkehrstechnik  
TU Darmstadt



- THANK YOU
- DANKE