The Art and Science of Transportation Research in the AI Era

A Gentle Introduction to SQL

M.Sc. Hiba Karam

Learning Goals

#1 Undersatnd what is a database and most used types

#2 Understand what is a relational database

#3 How to retrieve data from a SQL database

#4 Further write Basic SQL syntax

Lecture Structure

#1 About Database

#2 Relational Database

#3 SQL 101

#1 About Database

What is a Database?

An organized collection of structured data, stored electronically.
Managed by a Database Management System (DBMS).

Database System

The combination of database + DBMS + applications.

Why are Databases Important?

Efficiently handle **large and diverse** amounts of data.
Allow data to be **stored systematically** and **easily retrieved, filtered, sorted, and updated** accurately.

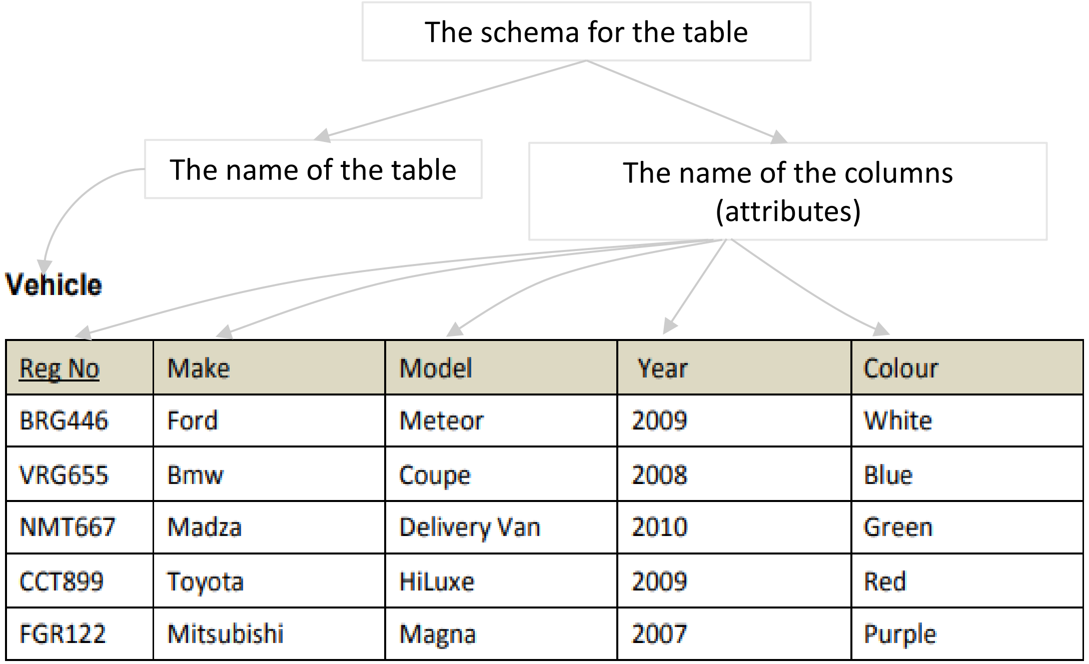
#2 Relational Database

Relational Databases
Organize data into **tables** with **rows** and **columns**.

Tables are grouped in **schemas** (containers/namespaces inside a database).
For **every column**, the schema defines the **allowed values** (e.g. Year is an **integer**).

This set of allowed values is the column’s **domain** or **data type**.
Structure is crucial: every row (e.g. each car) has the **same attributes/columns** - that’s the **schema**.
The **schema stays fixed**, but **rows change** as we add or update data.

SQL (Structured Query Language) manages data in relational databases.
Invented in **1974 at IBM** and still the **standard way** to access relational data.



The schema for the table

The name of the table

The name of the columns (attributes)

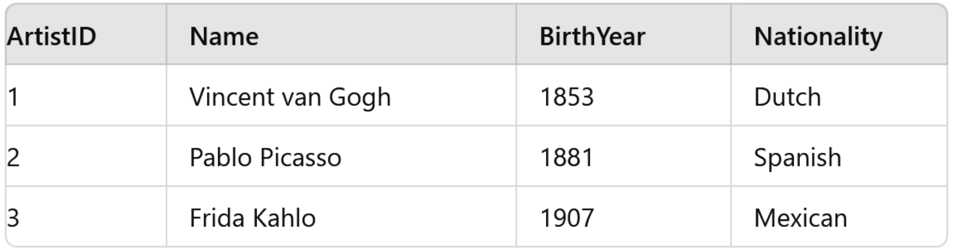
Vehicle

Reg No	Make	Model	Year	Colour
BRG446	Ford	Meteor	2009	White
VRG655	Bmw	Coupe	2008	Blue
NMT667	Madza	Delivery Van	2010	Green
CCT899	Toyota	HiLuxe	2009	Red
FGR122	Mitsubishi	Magna	2007	Purple

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

#2 Relational Database

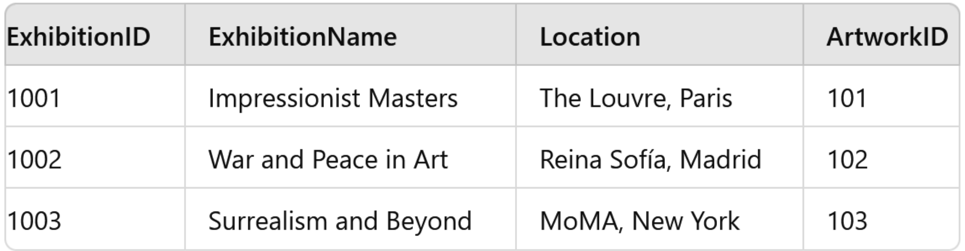
Table: Artists



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

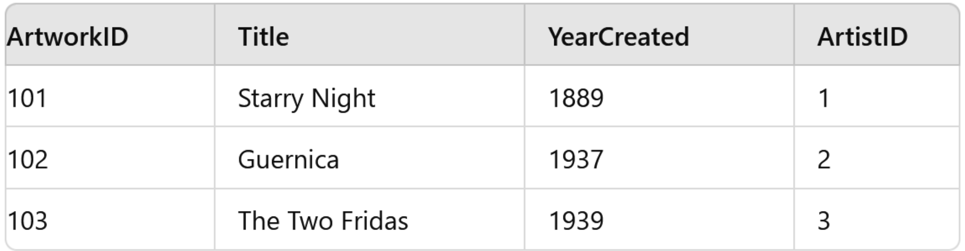


ExhibitionID	ExhibitionName	Location	ArtworkID
1001	Impressionist Masters	The Louvre, Paris	101
1002	War and Peace in Art	Reina Sofía, Madrid	102
1003	Surrealism and Beyond	MoMA, New York	103

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

Only one per table.

#2 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.

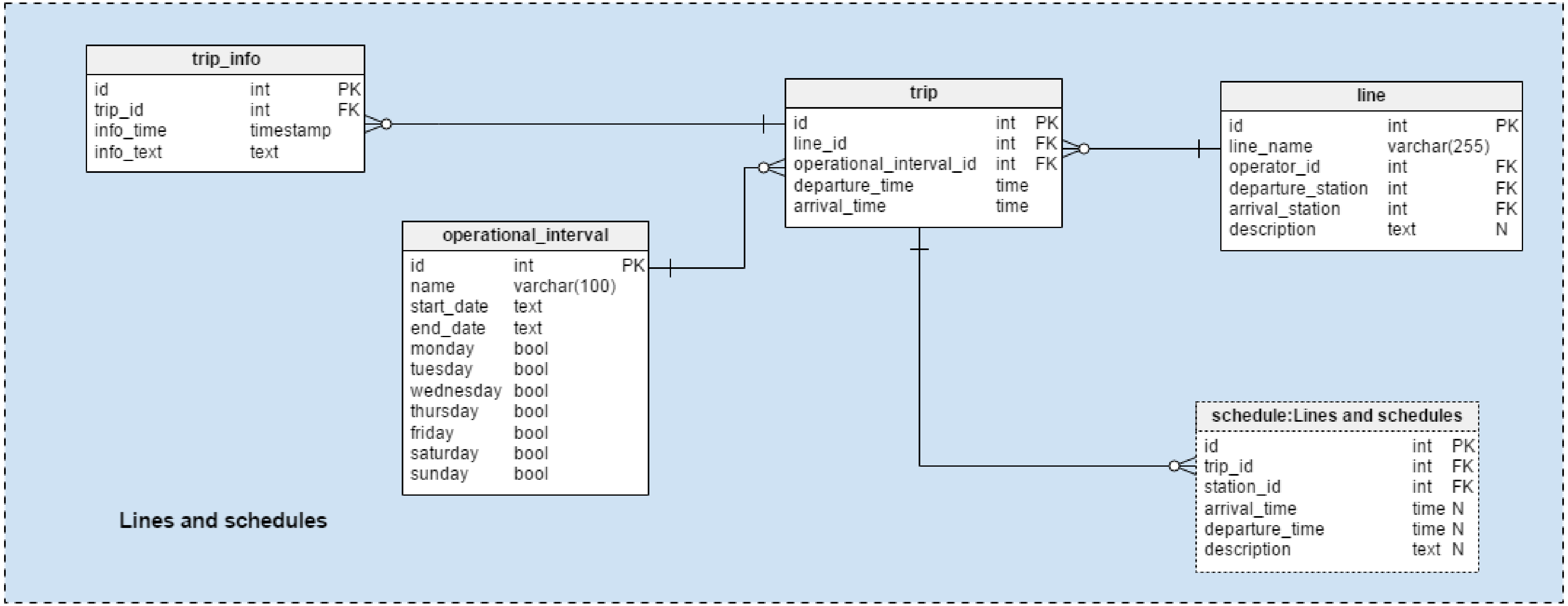
The action of creating and joining the tables…etc is done using SQL!

#2 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.**



trip_info

id	int	PK
trip_id	int	FK
info_time	timestamp	
info_text	text	

trip

id	int	PK
line_id	int	FK
operational_interval_id	int	FK
departure_time	time	
arrival_time	time	

line

id	int	PK
line_name	varchar(255)	
operator_id	int	FK
departure_station	int	FK
arrival_station	int	FK
description	text	N

operational_interval

id	int	PK
name	varchar(100)	
start_date	text	
end_date	text	
monday	bool	
tuesday	bool	
wednesday	bool	
thursday	bool	
friday	bool	
saturday	bool	
sunday	bool	

schedule:Lines and schedules

id	int	PK
trip_id	int	FK
station_id	int	FK
arrival_time	time	N
departure_time	time	N
description	text	N

Data model in a relational database

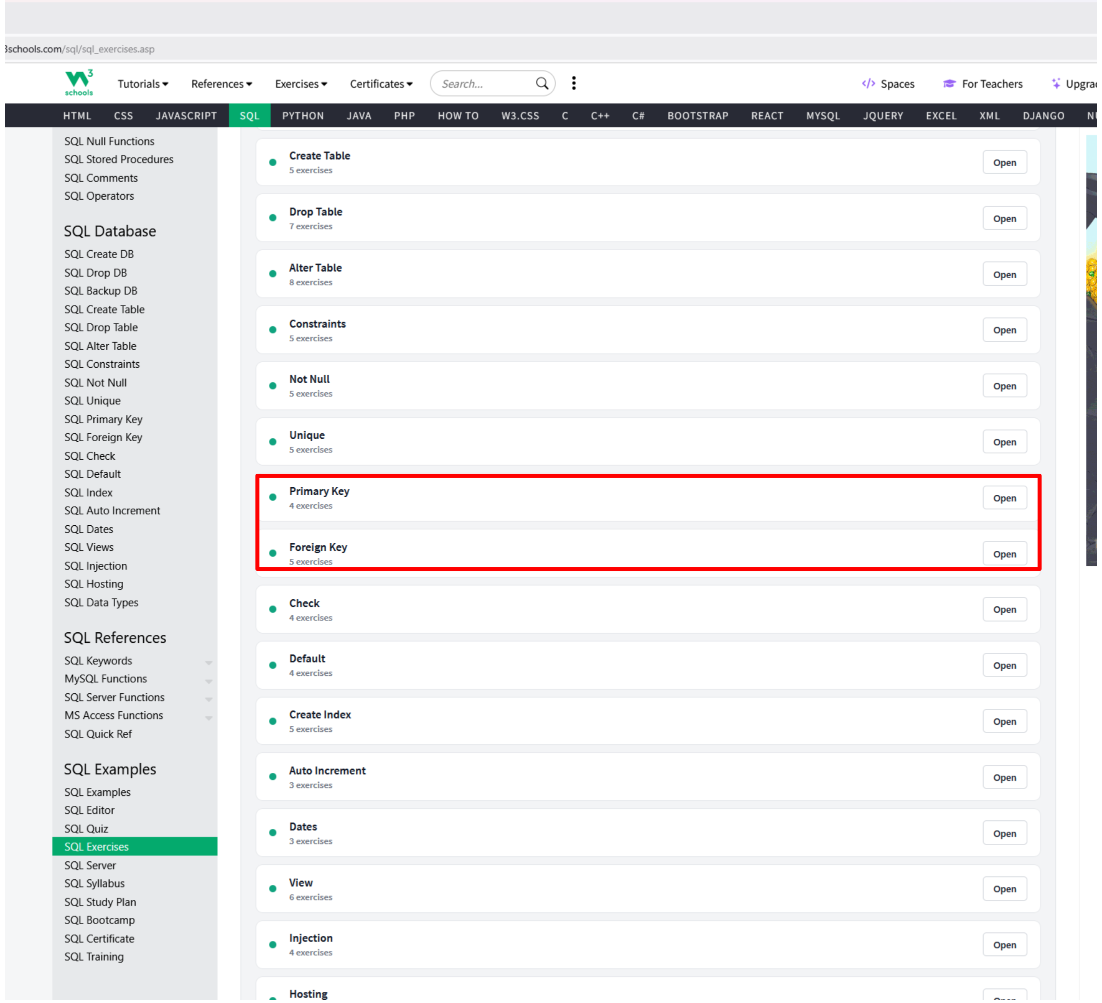
We can identify the correct tables to join, select the right columns, and apply filters more accurately, which **improves the performance and accuracy of data retrieval.**

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

#2 Relational Database

Go here: https://www.w3schools.com/sql/sql_exercises.asp

Only answer the first question for each.



Create Table
5 exercises
Open

Drop Table
7 exercises
Open

Alter Table
8 exercises
Open

Constraints
5 exercises
Open

Not Null
5 exercises
Open

Unique
5 exercises
Open

Primary Key
4 exercises
Open

Foreign Key
5 exercises
Open

Check
4 exercises
Open

Default
4 exercises
Open

Create Index
5 exercises
Open

Auto Increment
3 exercises
Open

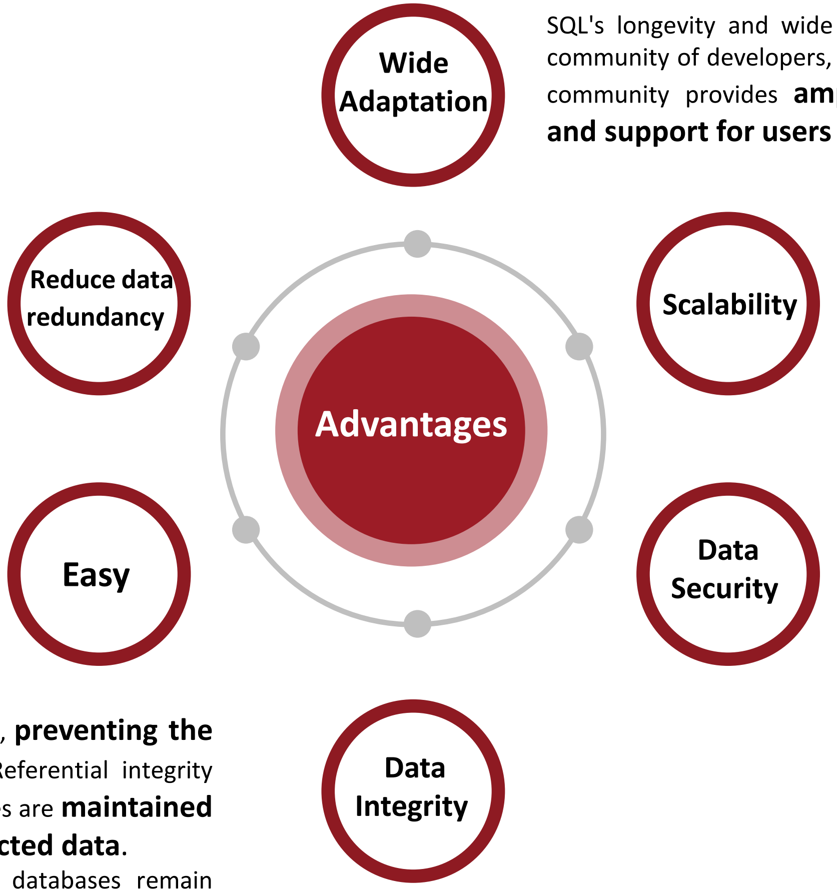
Dates
3 exercises
Open

View
6 exercises
Open

Injection
4 exercises
Open

Hosting

#2 Relational Database



SQL's longevity and wide adoption have resulted in a thriving community of developers, enthusiasts, and experts. The vast SQL community provides **ample resources, documentation, and support for users at all levels of expertise**

Optimized for handling large volumes of data and can scale to meet the demands

With features like encryption options and access controls, SQL databases **safeguard data from unauthorized access**

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

Easy data retrieval and manipulation

Store each fact once (e.g. station info in one table), then reuse it via keys instead of copying it everywhere.

#2 Relational Database

Why use relational databases in transportation planning and engineering?

Central, consistent storage of lines, stops, trips, schedules, passengers, sensors, etc.

Less redundancy, fewer errors each station, line, vehicle stored once and reused.

Easy data integration: join many tables for example GPS logs + timetable + ridership).

Powerful analysis with SQL: filter, aggregate, and compare scenarios quickly.

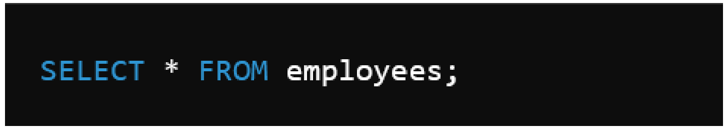
Scales and shares well: many users, large datasets, dashboards and models all use the same trusted data.

#2 Relational Database

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.

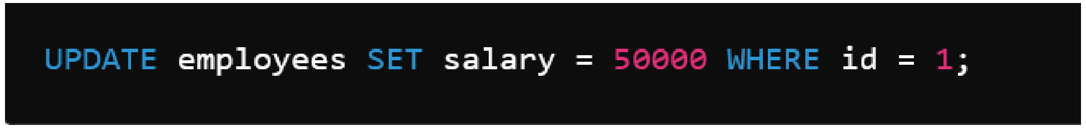
#3 SQL 101

- 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.



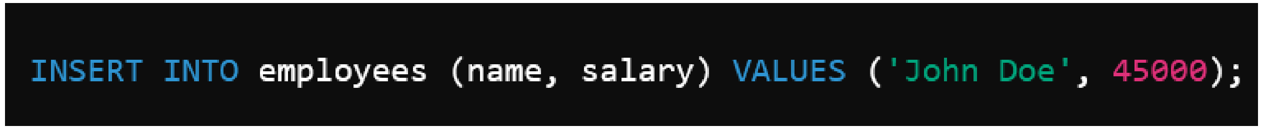
```sql
SELECT * FROM employees;
```

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



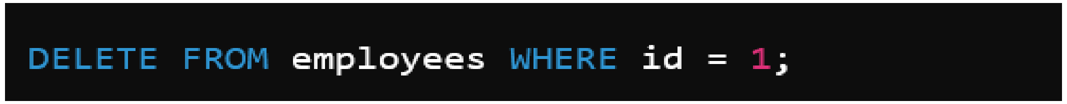
```sql
UPDATE employees SET salary = 50000 WHERE id = 1;
```

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



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

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



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

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

We will focus on SELECT to pull out specific data points.

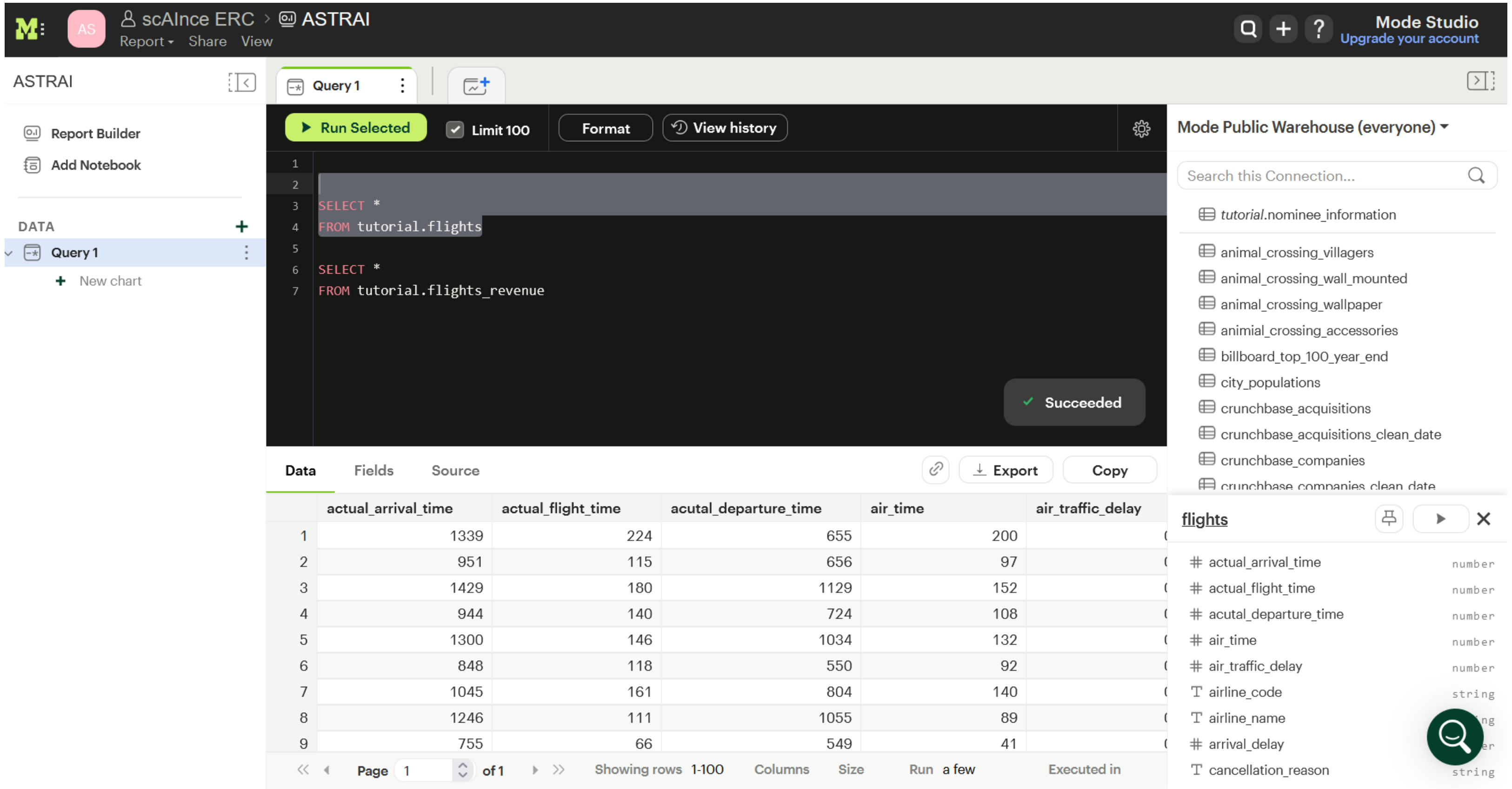
#3.1 SELECT FROM

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.**



```
SELECT *
FROM tutorial.flights

SELECT *
FROM tutorial.flights_revenue
```

	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	

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

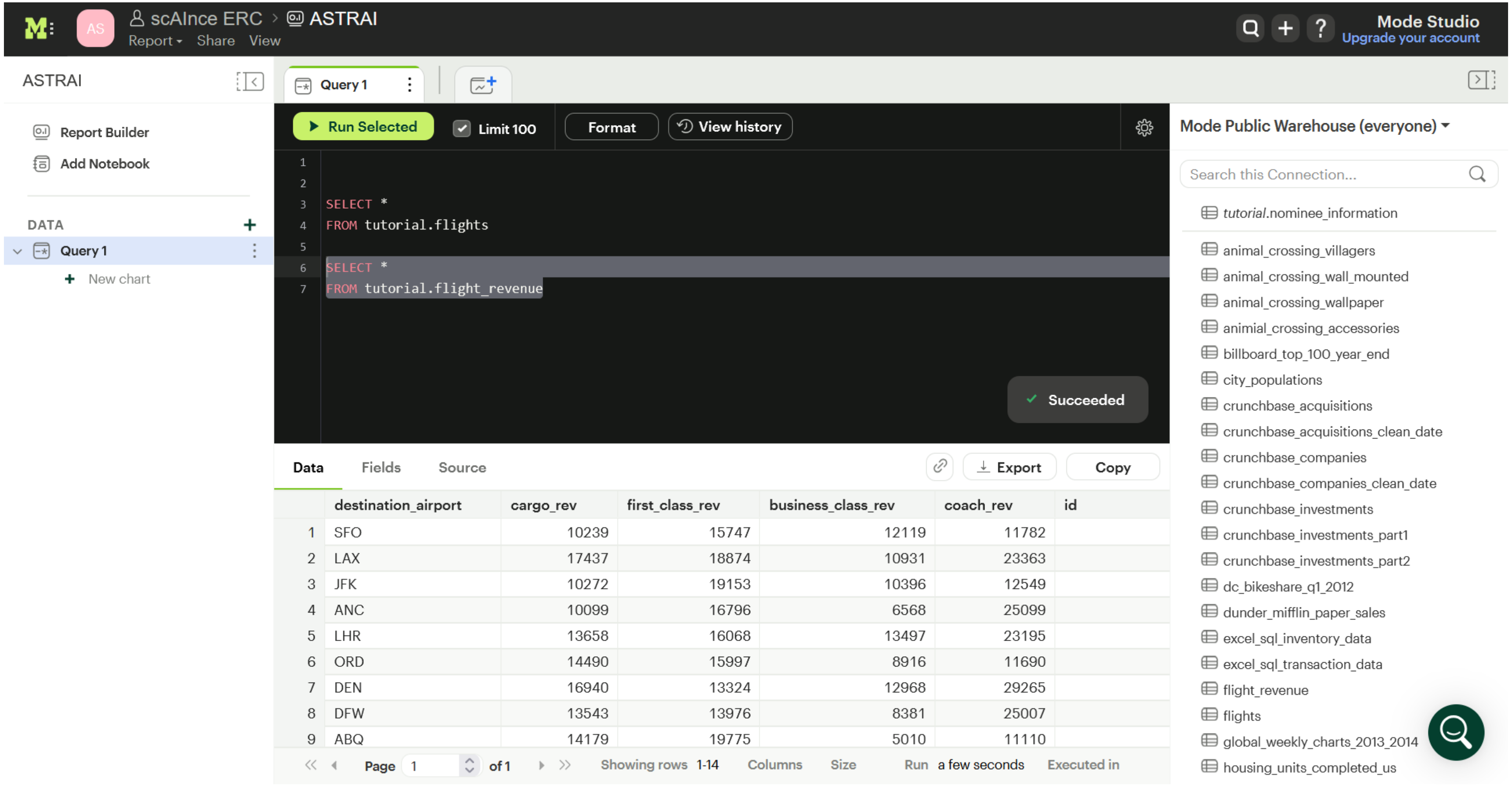
tutorial.nominee_information

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

flights

# actual_arrival_time	number
# actual_flight_time	number
# acutal_departure_time	number
# air_time	number
# air_traffic_delay	number
T airline_code	string
T airline_name	ng
# arrival_delay	er
T cancellation_reason	string

#3.1 SELECT FROM


scAInce ERC > ASTRAI
Report Share View
Mode Studio
Upgrade your account
ASTRAI
Report Builder
Add Notebook
DATA
Query 1
New chart
Query 1
Run Selected
☑ Limit 100
Format
View history
✓ Succeeded
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
Data Fields Source
Export
Copy
Page 1 of 1
Showing rows 1-14
Columns
Size
Run a few seconds
Executed in

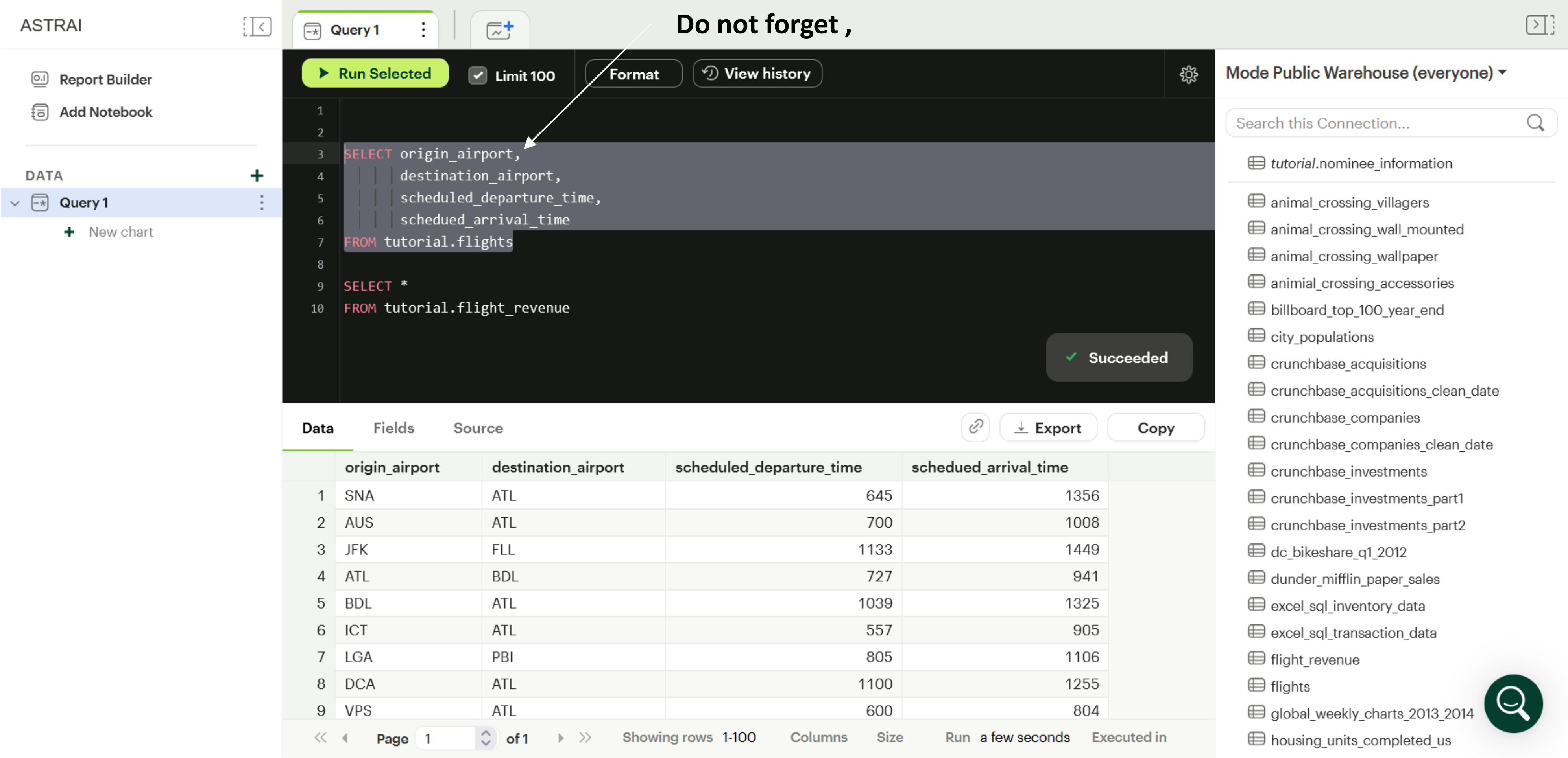
```
SELECT *
FROM tutorial.flights

SELECT *
FROM tutorial.flight_revenue
```

	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	

#3.1 SELECT FROM

You can also select a specific column/s

Do not forget ,

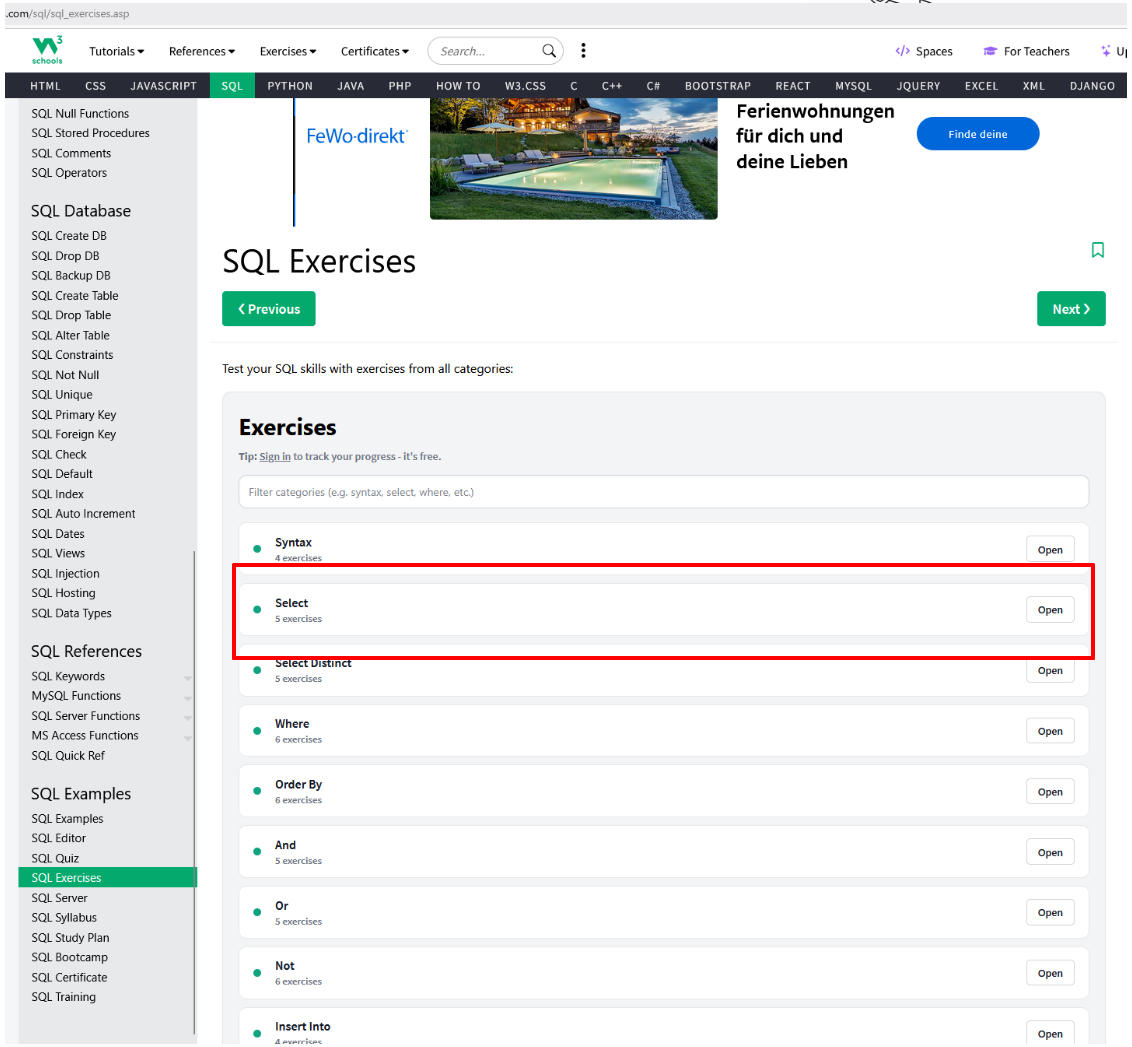


ASTRAI
Report Builder
Add Notebook
DATA
Query 1
New chart
Query 1
Run Selected
☑ Limit 100
Format
View history
Mode Public Warehouse (everyone)
Search this Connection...
tutorial.nominee_information
animal_crossing_villagers
animal_crossing_wall_mounted
animal_crossing_wallpaper
animial_crossing_accessories
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
Succeeded
Data
Fields
Source
Export
Copy
Page 1 of 1
Showing rows 1-100
Columns
Size
Run a few seconds
Executed in

```


SELECT origin_airport,
       destination_airport,
       scheduled_departure_time,
       schedued_arrival_time
FROM tutorial.flights

SELECT *
FROM tutorial.flight_revenue
```

	origin_airport	destination_airport	scheduled_departure_time	schedued_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

#3.1 SELECT FROM



SQL Exercises

Test your SQL skills with exercises from all categories:

Exercises

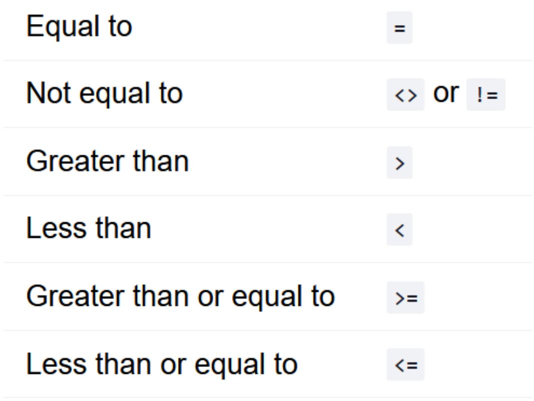
Tip: Sign in to track your progress - it's free.

Filter categories (e.g. syntax, select, where, etc.)

- Syntax – 4 exercises – Open
- Select – 5 exercises – Open
- Select Distinct – 5 exercises – Open
- Where – 6 exercises – Open
- Order By – 6 exercises – Open
- And – 5 exercises – Open
- Or – 5 exercises – Open
- Not – 6 exercises – Open
- Insert Into – 4 exercises – Open

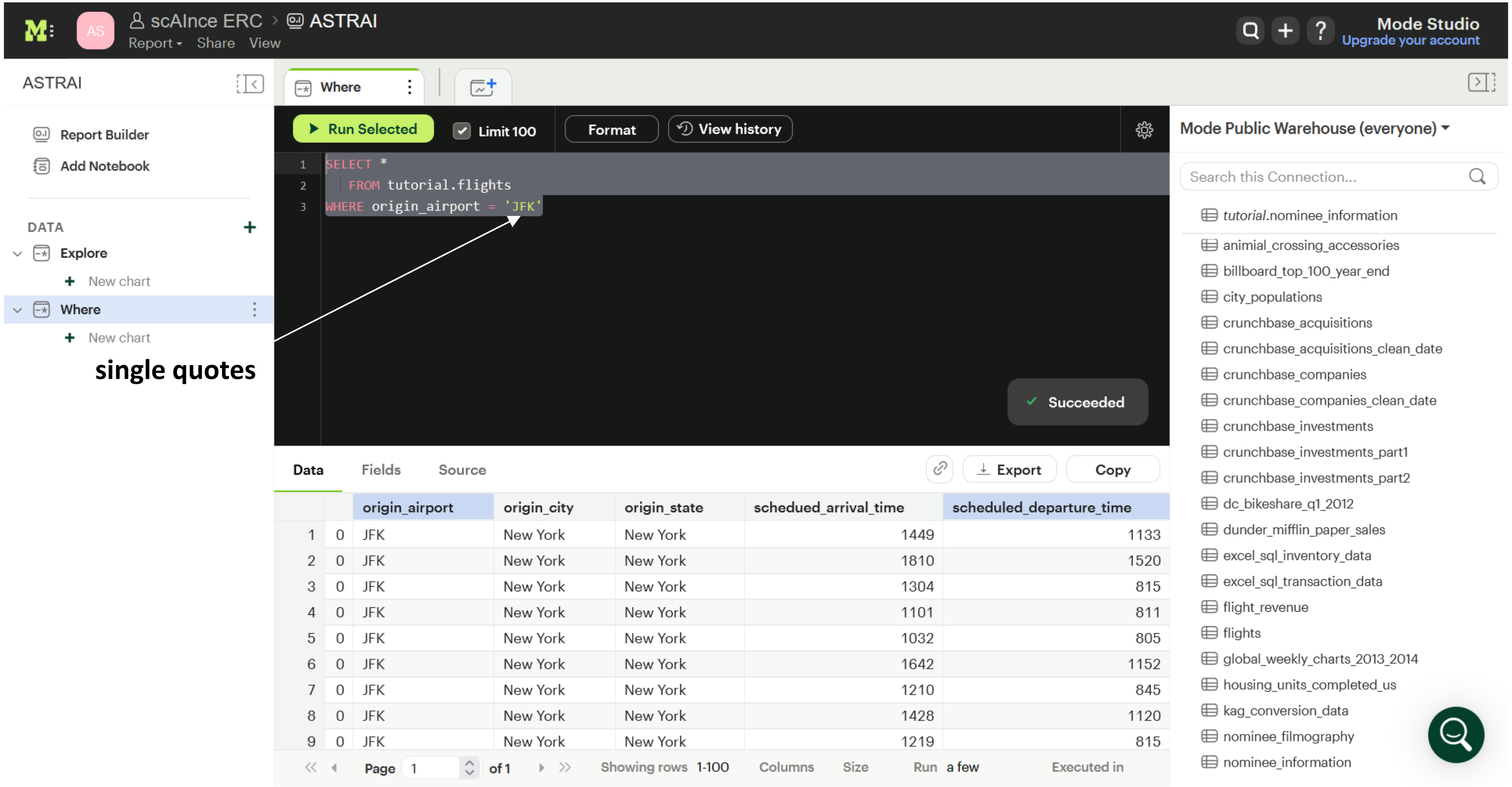
#3.2 WHERE Filter

- 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



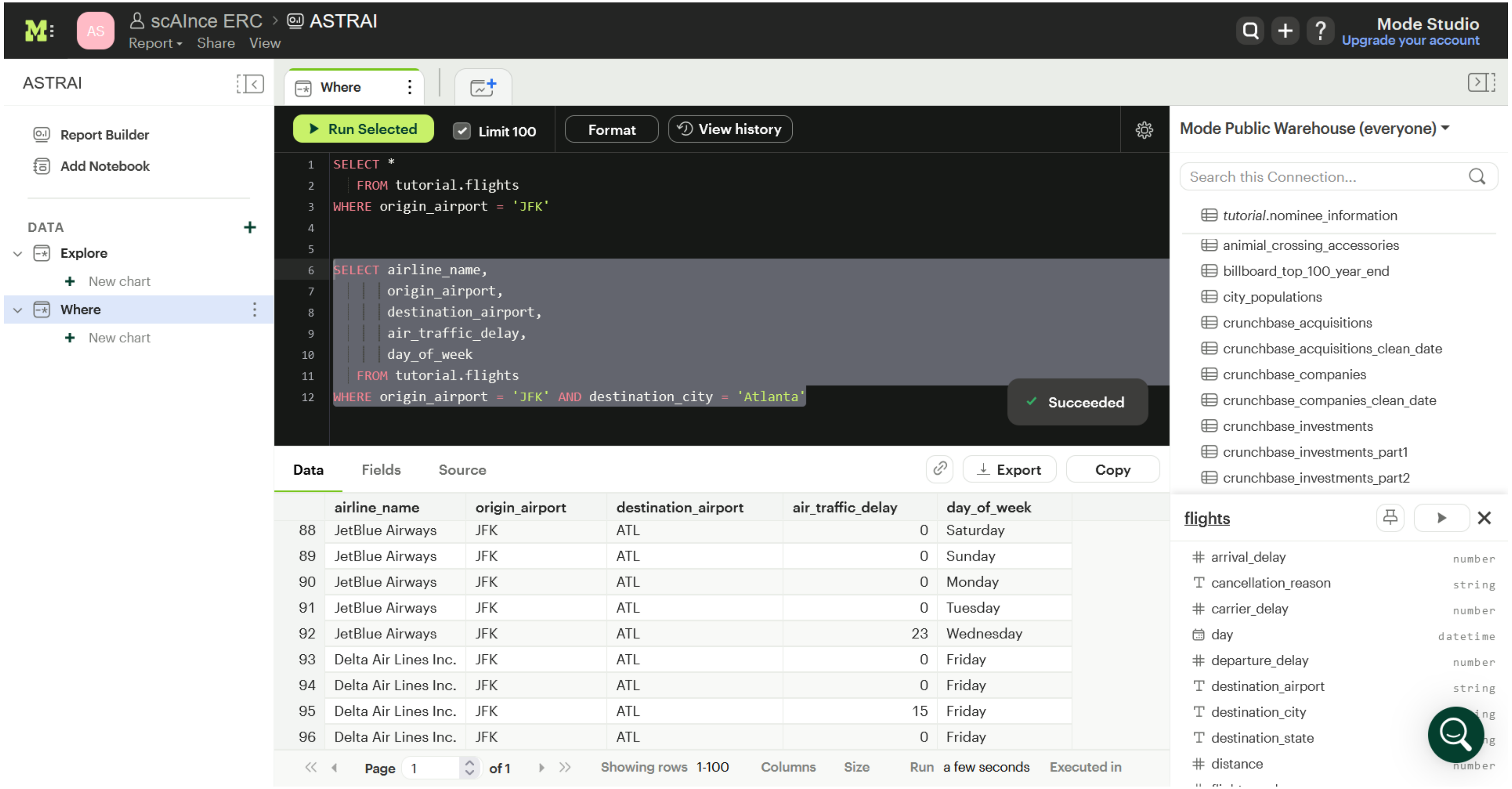
```sql
SELECT *
  FROM tutorial.flights
WHERE origin_airport = 'JFK'
```

single quotes

		origin_airport	origin_city	origin_state	schedued_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

#3.2 WHERE Filter

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



```
SELECT *
  FROM tutorial.flights
WHERE origin_airport = 'JFK'


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

	airline_name	origin_airport	destination_airport	air_traffic_delay	day_of_week
88	JetBlue Airways	JFK	ATL	0	Saturday
89	JetBlue Airways	JFK	ATL	0	Sunday
90	JetBlue Airways	JFK	ATL	0	Monday
91	JetBlue Airways	JFK	ATL	0	Tuesday
92	JetBlue Airways	JFK	ATL	23	Wednesday
93	Delta Air Lines Inc.	JFK	ATL	0	Friday
94	Delta Air Lines Inc.	JFK	ATL	0	Friday
95	Delta Air Lines Inc.	JFK	ATL	15	Friday
96	Delta Air Lines Inc.	JFK	ATL	0	Friday

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

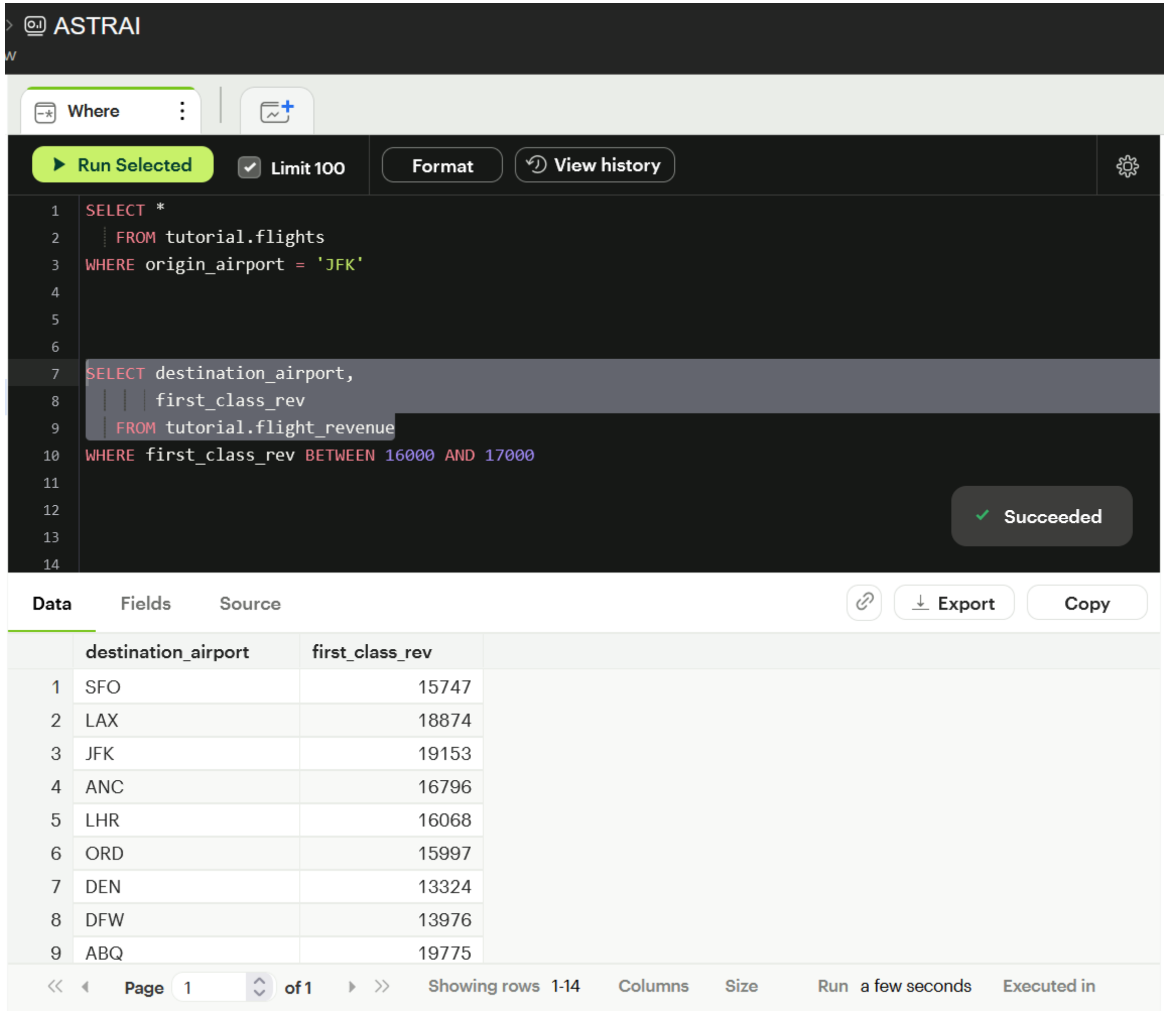
#3.2 WHERE Filter

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.

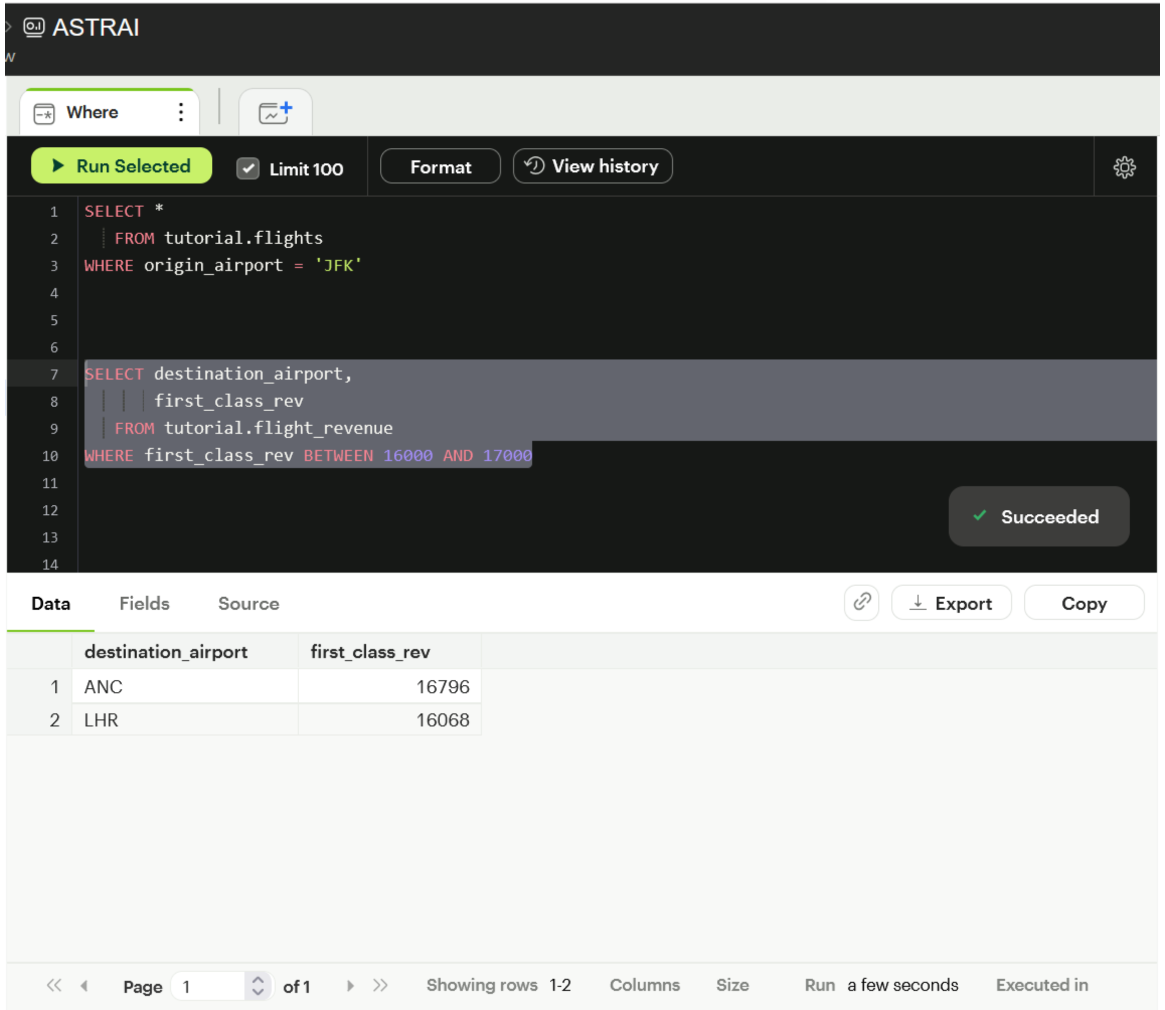
What is the difference between zero and null value?

#3.2 WHERE Filter



```sql
SELECT *
  FROM tutorial.flights
WHERE origin_airport = 'JFK'



SELECT destination_airport,
       first_class_rev
  FROM tutorial.flight_revenue
WHERE first_class_rev BETWEEN 16000 AND 17000
```

	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



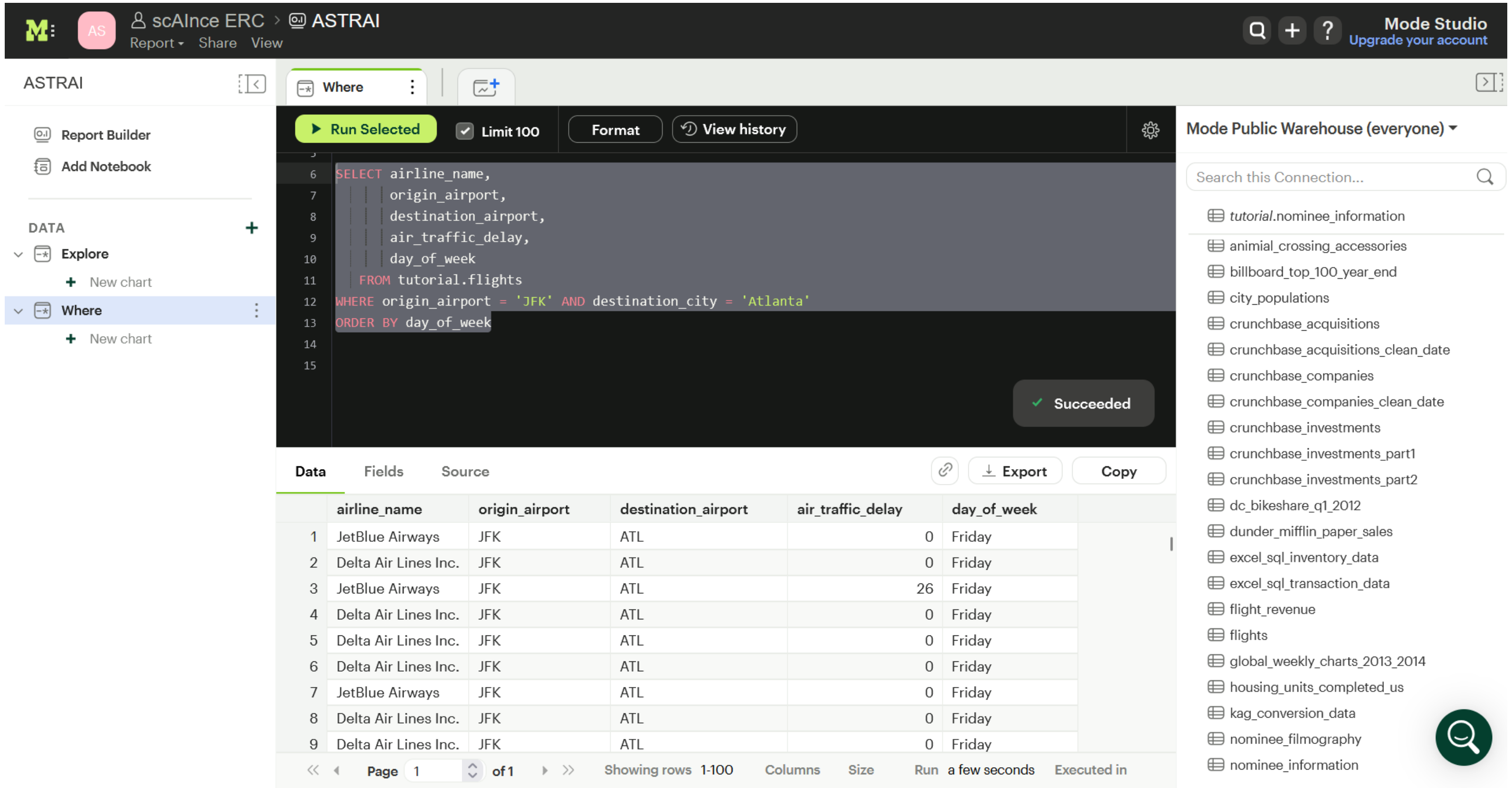
```sql
SELECT *
  FROM tutorial.flights
WHERE origin_airport = 'JFK'



SELECT destination_airport,
       first_class_rev
  FROM tutorial.flight_revenue
WHERE first_class_rev BETWEEN 16000 AND 17000
```

	destination_airport	first_class_rev
1	ANC	16796
2	LHR	16068

What is the difference between these two queries?

#3.3 ORDER BY Sorting

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.



```
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
```

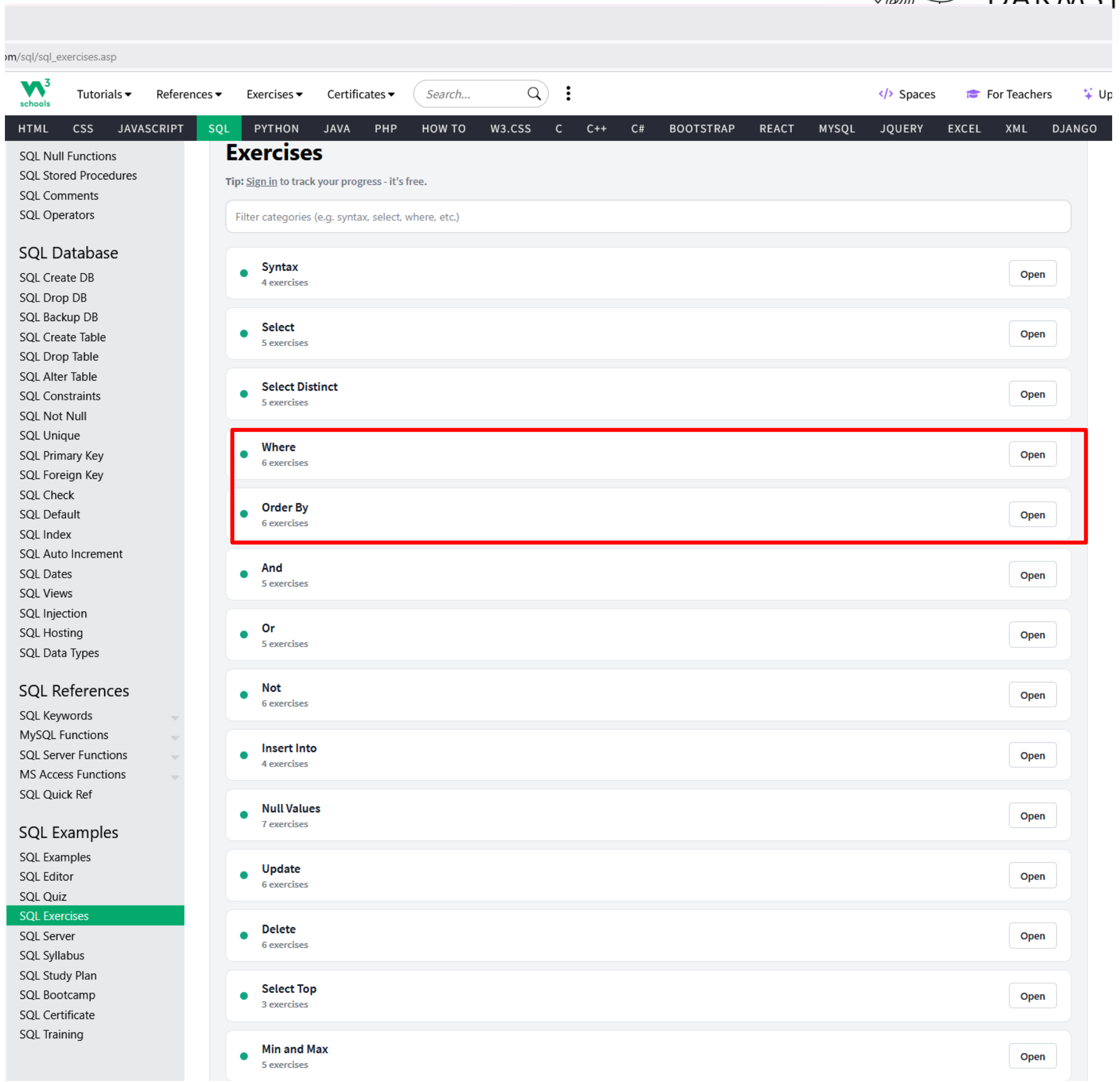
	airline_name	origin_airport	destination_airport	air_traffic_delay	day_of_week
1	JetBlue Airways	JFK	ATL	0	Friday
2	Delta Air Lines Inc.	JFK	ATL	0	Friday
3	JetBlue Airways	JFK	ATL	26	Friday
4	Delta Air Lines Inc.	JFK	ATL	0	Friday
5	Delta Air Lines Inc.	JFK	ATL	0	Friday
6	Delta Air Lines Inc.	JFK	ATL	0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

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

tutorial.nominee_information
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
kag_conversion_data
nominee_filmography
nominee_information

What changed?

#3.3 Filter



om/sql/sql_exercises.asp

Tutorials ▾ References ▾ Exercises ▾ Certificates ▾ Search... Spaces For Teachers Up
HTML CSS JAVASCRIPT SQL PYTHON JAVA PHP HOW TO W3.CSS C C++ C# BOOTSTRAP REACT MYSQL JQUERY EXCEL XML DJANGO

SQL Null Functions
SQL Stored Procedures
SQL Comments
SQL Operators

SQL Database

SQL Create DB
SQL Drop DB
SQL Backup DB
SQL Create Table
SQL Drop Table
SQL Alter Table
SQL Constraints
SQL Not Null
SQL Unique
SQL Primary Key
SQL Foreign Key
SQL Check
SQL Default
SQL Index
SQL Auto Increment
SQL Dates
SQL Views
SQL Injection
SQL Hosting
SQL Data Types

SQL References

SQL Keywords
MySQL Functions
SQL Server Functions
MS Access Functions
SQL Quick Ref

SQL Examples

SQL Examples
SQL Editor
SQL Quiz
SQL Exercises
SQL Server
SQL Syllabus
SQL Study Plan
SQL Bootcamp
SQL Certificate
SQL Training

Exercises

Tip: Sign in to track your progress - it's free.

Filter categories (e.g. syntax, select, where, etc.)

●	Syntax 4 exercises	Open
●	Select 5 exercises	Open
●	Select Distinct 5 exercises	Open
●	Where 6 exercises	Open
●	Order By 6 exercises	Open
●	And 5 exercises	Open
●	Or 5 exercises	Open
●	Not 6 exercises	Open
●	Insert Into 4 exercises	Open
●	Null Values 7 exercises	Open
●	Update 6 exercises	Open
●	Delete 6 exercises	Open
●	Select Top 3 exercises	Open
●	Min and Max 5 exercises	Open

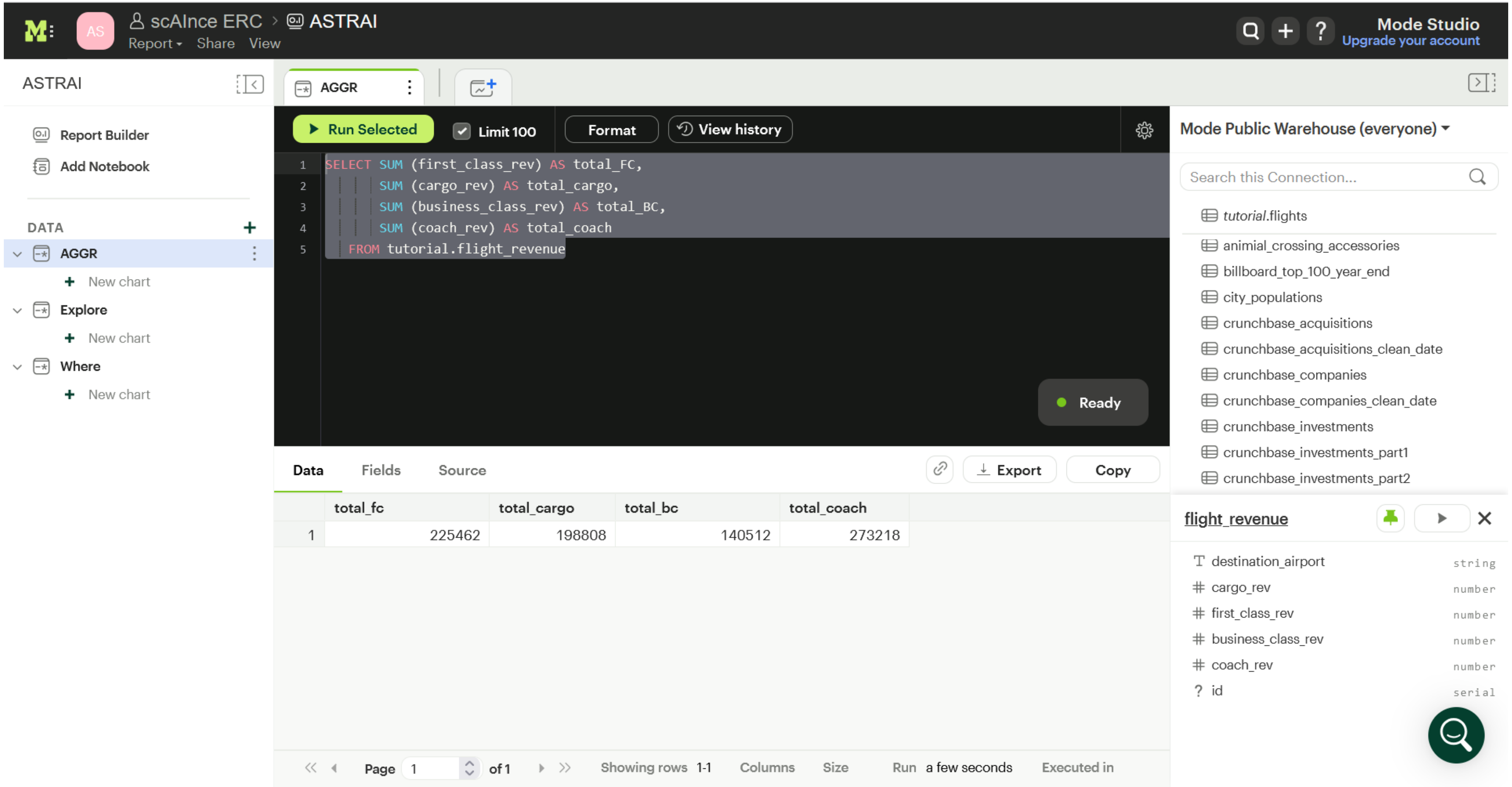
#3.4 AGGREGATION FUNCTIONS

- 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



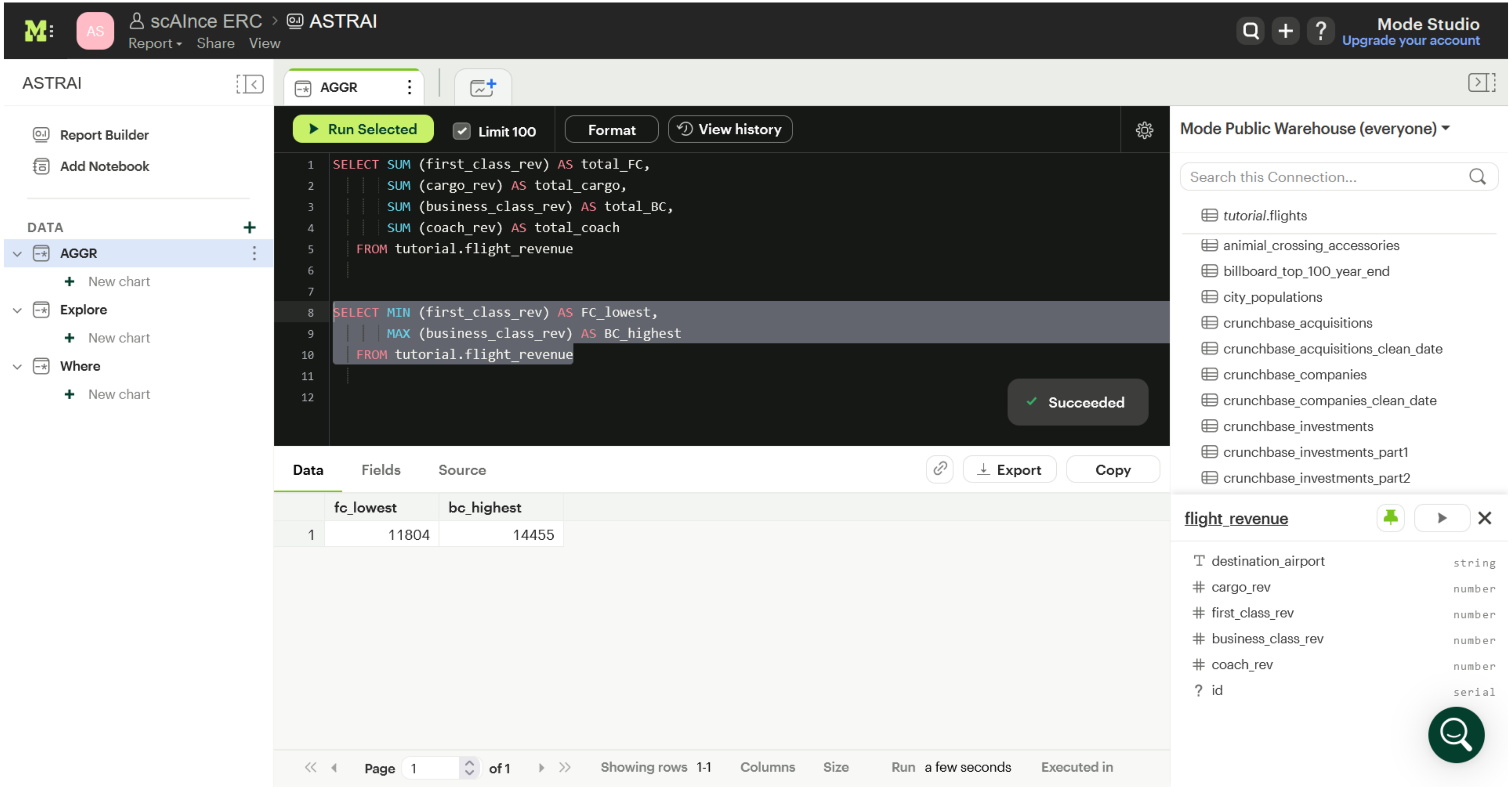
```sql
SELECT SUM (first_class_rev) AS total_FC,
       SUM (cargo_rev) AS total_cargo,
       SUM (business_class_rev) AS total_BC,
       SUM (coach_rev) AS total_coach
  FROM tutorial.flight_revenue
```

	total_fc	total_cargo	total_bc	total_coach
1	225462	198808	140512	273218

flight_revenue

destination_airport	string
cargo_rev	number
first_class_rev	number
business_class_rev	number
coach_rev	number
id	serial

#3.4 AGGREGATION FUNCTIONS



```sql
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
```

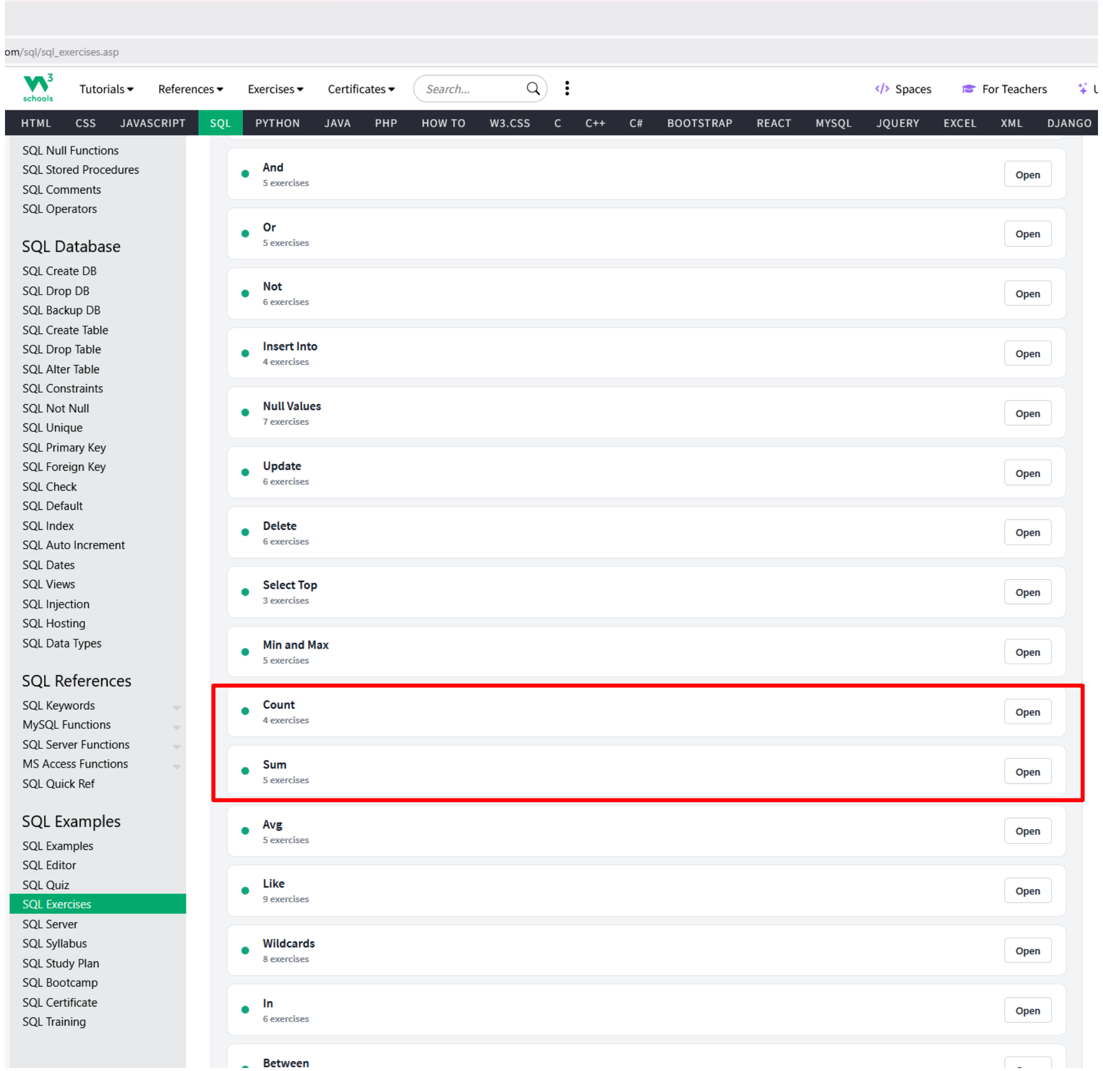
	fc_lowest	bc_highest
1	11804	14455

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

#3.4 AGGREGATION FUNCTIONS



● And 5 exercises Open

● Or 5 exercises Open

● Not 6 exercises Open

● Insert Into 4 exercises Open

● Null Values 7 exercises Open

● Update 6 exercises Open

● Delete 6 exercises Open

● Select Top 3 exercises Open

● Min and Max 5 exercises Open

● Count 4 exercises Open

● Sum 5 exercises Open

● Avg 5 exercises Open

● Like 9 exercises Open

● Wildcards 8 exercises Open

● In 6 exercises Open

● Between

- **THANK YOU**
- **DANKE**