

Bootcamp Live 02

Intro to Databases with



Instructor: Kasidis Satangmongkol





Why should we learn databases?



Spreadsheets



Databases



Data Studio

Dashboard



What can databases do?



Store



Analyze



Present

Some databases also
have data visualization
features



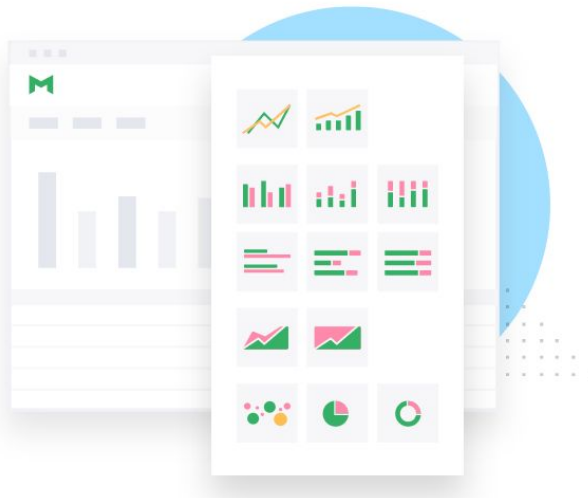
SQL is a fundamental skill for data analyst

The screenshot shows the Mode SQL interface. At the top, there's a header with 'Sales - Account Metrics' and 'Account Breakdown'. Below that, a 'Query1' tab is active, showing a SQL query. The query is a SELECT statement that joins 'user_accounts' and 'user_profiles' tables, filtering for 'complete' profiles. Below the query editor, there's a 'Run' button and a 'Limit 100' checkbox. The results section shows a table with 1,298,420 rows and 9GB of data returned in 45 seconds. The table has columns: account_name, signup_date, profile_status, and user_id. The first few rows are visible, showing accounts like 'Acadia Realty Trust' and 'Reven Housing REIT, Inc.'.

```
1 SELECT us.id AS user_id,
2       CASE WHEN p.created_at IS NOT NULL THEN 'complete'
3       ELSE 'incomplete' END AS profile_status,
4       us.created_at AS started_at,
5       p.created_at AS completed_at
6 FROM ((@user_accounts AS us ))
7 LEFT JOIN user_profiles p
8 ON p.user_id = user.id
```

	account_name	signup_date	profile_status	user_id
1	Acadia Realty Trust	2019-05-15T16:12:15.000Z	complete	4288
2	Reven Housing REIT, Inc.	2019-05-15T14:14:13.000Z	complete	10208
3	Horizon Pharma plc	2019-05-15T14:06:16.000Z	complete	6121
4	Emclaire Financial Corp	2019-05-15T13:44:37.000Z	complete	209
5	Cushing Renaissance Fund (The)	2019-05-15T12:40:21.000Z	complete	1982
6	Brasilian Cia Brasileira De Beneficencia	2019-05-15T12:34:46.000Z	complete	889

Showing 1-100 of 1,298,420 results



Customize your visualizations

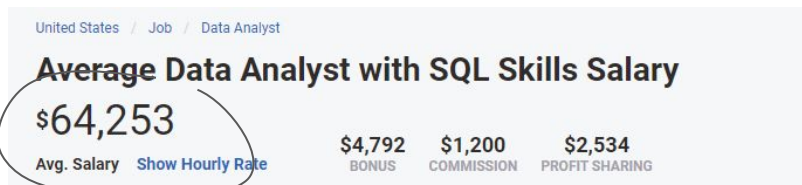
Visualize any SQL result with Mode's native chart builder and add a personal touch with our HTML editor.

SQL is easy to learn, Quick win!



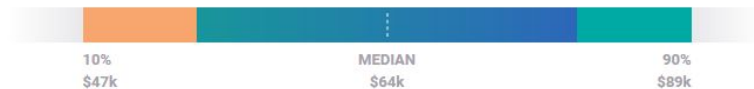
Average salary in USA

Data Analyst with SQL Skills Salary | PayScale



Convert to THB
* 30 / 2.5

The average salary for a Data Analyst with SQL skills is \$64,253.



You can find your net worth using this free tool

Is Data Analyst your job title? Find out what you should be paid
Use our tool to get a personalized report on your market worth. [What's this?](#)

Location:

Thailand [\(change\)](#)

Years in Field/Career:

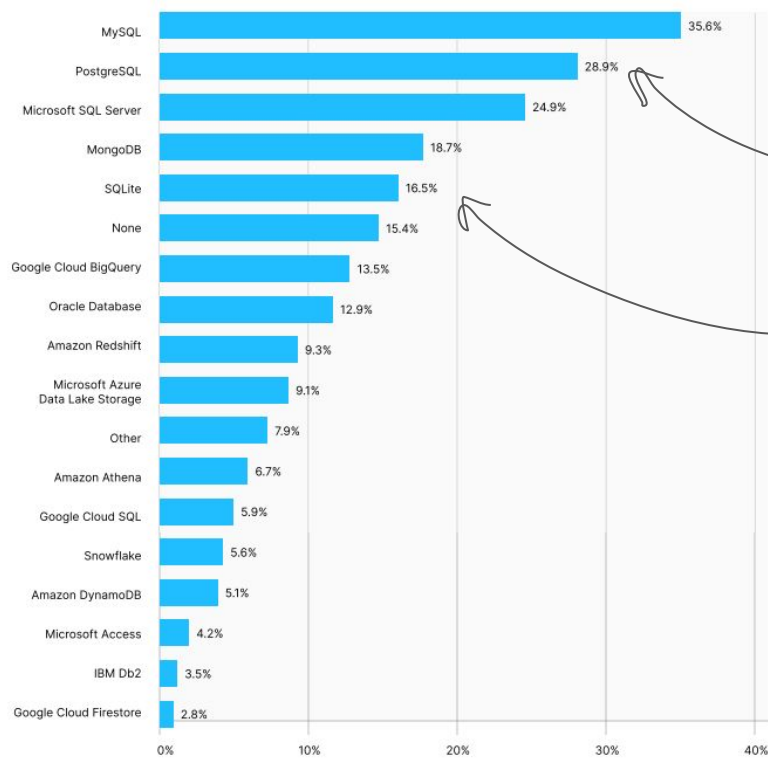
[Find your market worth »](#)

How it works:

- 1 Enter city & years of experience
- 2 Add pay factors like skills & education
- 3 Find your market worth with a report tailored to you



There are many versions of SQL

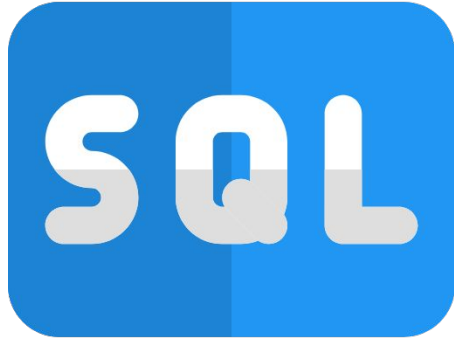


We'll teach PostgreSQL too :)

SQLite is a standard language (pure)



We have been using SQL for 50 years



Structured Query Language



verb. Ask a question about our data



How database work?



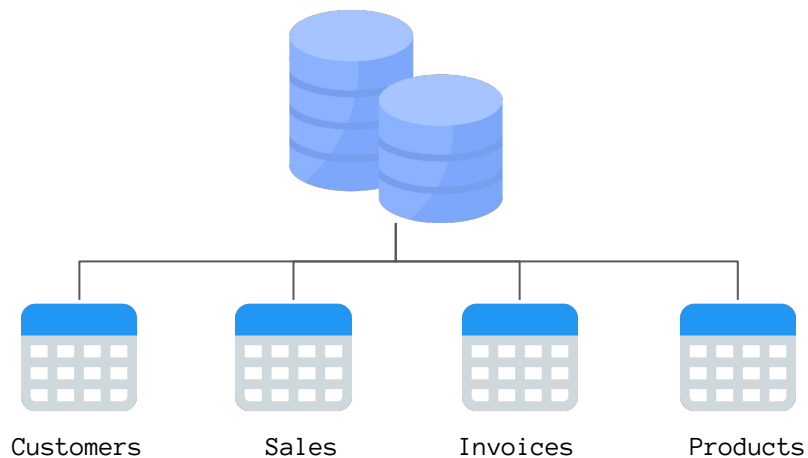
Every company has databases
(99.99%)

Query
←
→
Result





What database looks like?



You are already familiar with database technology

employee_csv

Schema Details Preview

Row	EmployeeID	FirstName	LastName	HireDate	City	Salary	Terminated
1	1	Sun	Bak	2020-01-28	Seoul	120000	false
2	3	Kala	Dandekar	2020-03-31	Mumbai	140000	false
3	4	Riley	Blue	2020-04-18	London	90000	false
4	6	Lito	Rodriguez	2020-06-19	Mexico City	250000	false
5	2	Nomi	Marks	2020-02-11	San Francisco	110000	true
6	5	Wolfgang	Bogdanow	2020-05-20	Berlin	115000	true
7	7	Will	Gorski	2020-07-04	Chicago	180000	true



At the heart of SQL is selecting the data you want

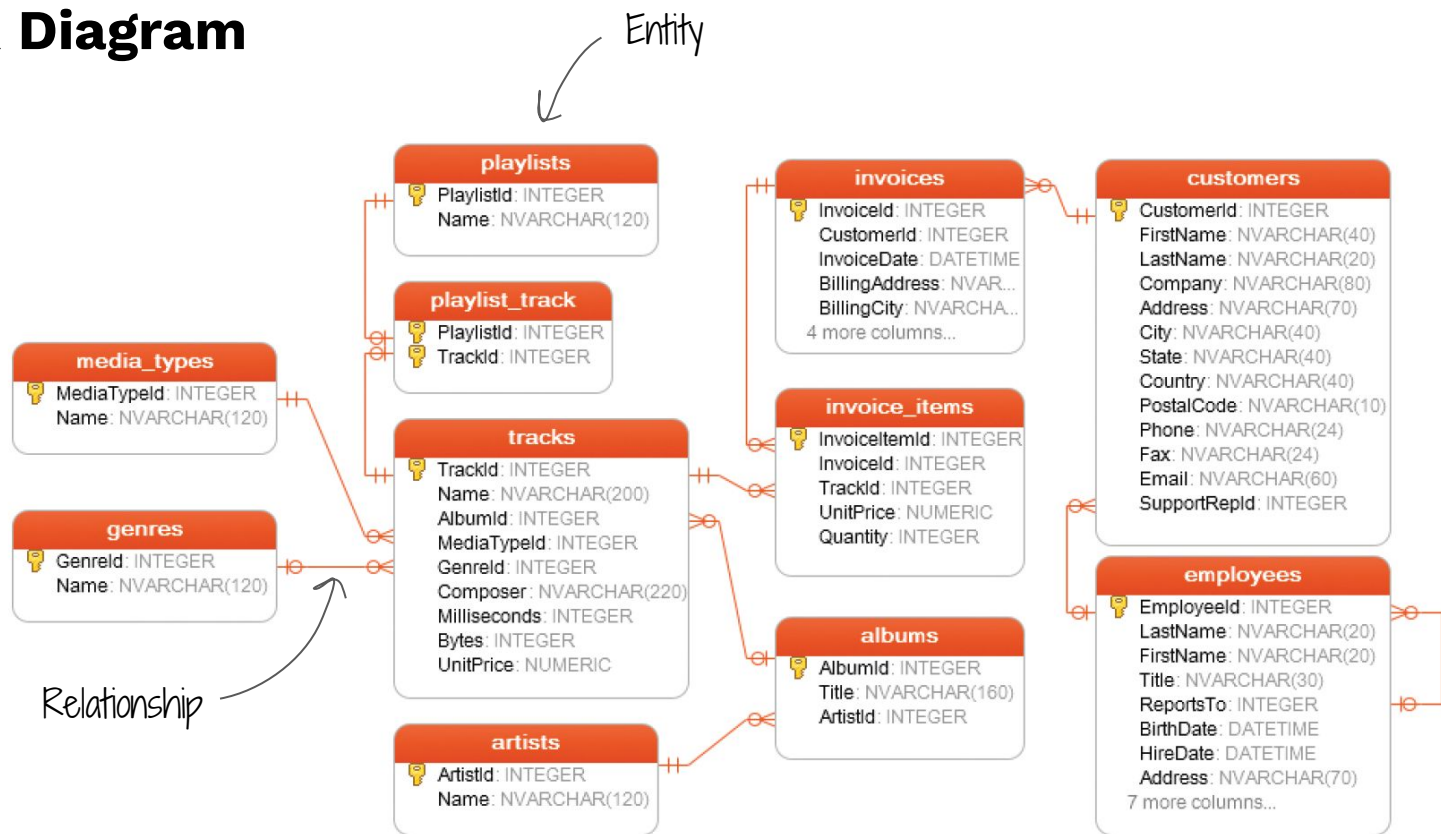
Select columns

Row	EmployeeID	FirstName	LastName	HireDate	City	Salary	Terminated
1	1	Sun	Bak	2020-01-28	Seoul	120000	false
2	3	Kala	Dandekar	2020-03-31	Mumbai	140000	false
3	4	Riley	Blue	2020-04-18	London	90000	false
4	6	Lito	Rodriguez	2020-06-19	Mexico City	250000	false
5	2	Nomi	Marks	2020-02-11	San Francisco	110000	true
6	5	Wolfgang	Bogdanow	2020-05-20	Berlin	115000	true
7	7	Will	Gorski	2020-07-04	Chicago	180000	true

Filter rows

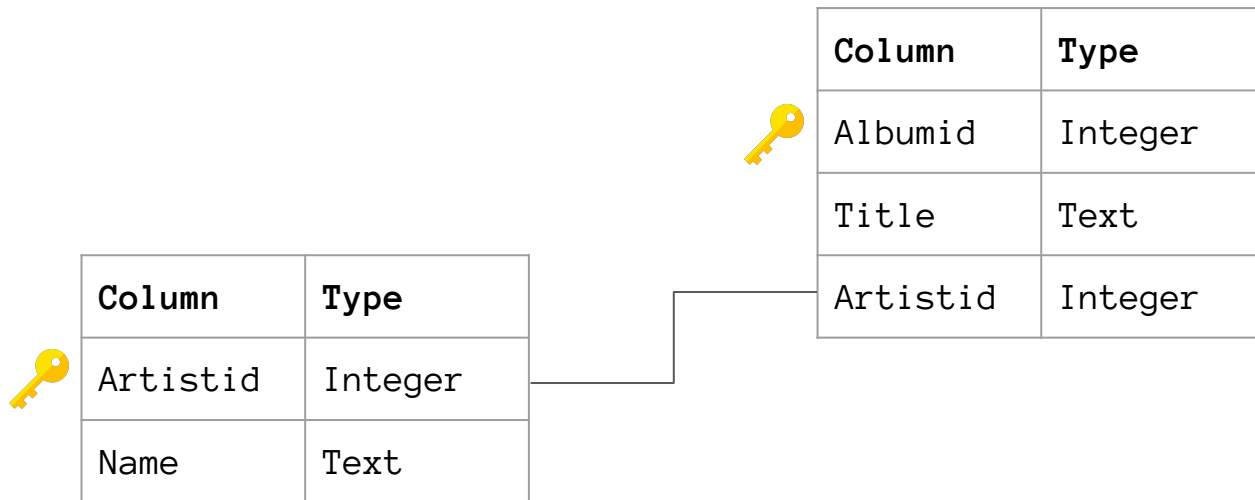


ER Diagram





Primary and foreign keys





SQL clauses we use in our data analyst role

Clauses	What it does?
SELECT	Select columns
FROM	From table
JOIN	Join multiple tables
WHERE	Filter data
Aggregate Functions	AVG SUM MIN MAX COUNT
GROUP BY	Group by statistics
HAVING	Filter groups
ORDER BY	Sort data



The first part will cover all the SQL basics :)





Select all columns

```
SELECT * FROM customers;
```



Select all columns

SELECT * **FROM** customers;

Upper case

Table name

Close with ;



Select specific columns

SELECT

 firstname,

 lastname,

 email,

 country

FROM customers;



Choose specific columns



Filter rows with where clause

```
SELECT *  
FROM customers  
WHERE country = 'USA';
```

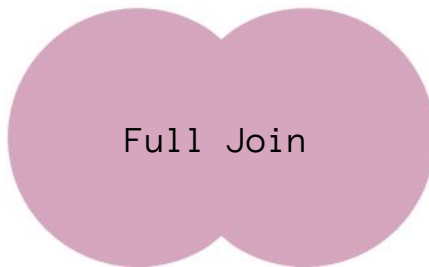
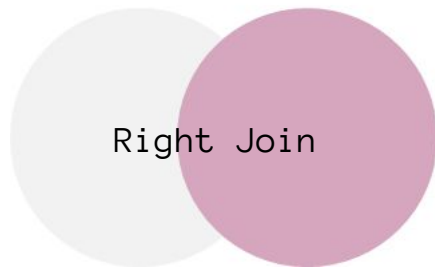
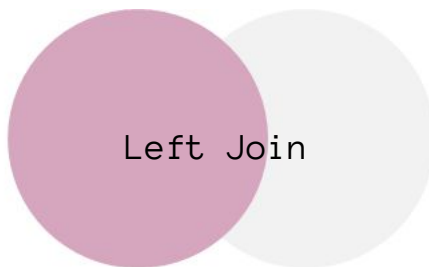
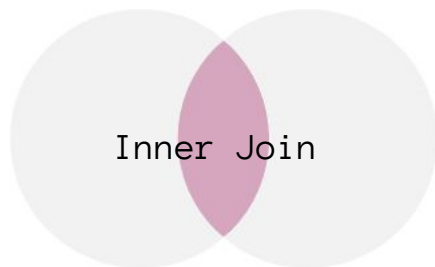
condition



```
SELECT *  
FROM customers  
WHERE country IN ('USA', 'Canada', 'United Kingdom');
```



Review join types



Inner and Left Join contribute around 90-95% of our work



Inner join

Table 1

PK_ID	Name
1	David
2	John
3	Marry
4	Anna
5	Kevin

+

Table 2

FK_ID	Major
1	Econ
2	Econ
5	Data
12	Engineer
35	Mkt

=

Result Set

PK_ID	Name	Major
1	David	Econ
2	John	Econ
5	Kevin	Data



Left join

Table 1

PK_ID	Name
1	David
2	John
3	Marry
4	Anna
5	Kevin

+

Table 2

FK_ID	Major
1	Econ
2	Econ
5	Data
12	Engineer
35	Mkt

=

Result Set

PK_ID	Name	Major
1	David	Econ
2	John	Econ
3	Marry	NULL
4	Anna	NULL
5	Kevin	Data



Full join

Table 1

PK_ID	Name
1	David
2	John
3	Marry
4	Anna
5	Kevin

+

Table 2

FK_ID	Major
1	Econ
2	Econ
5	Data
12	Engineer
35	Mkt

=

Result Set

PK_ID	Name	Major
1	David	Econ
2	John	Econ
3	Mary	NULL
4	Anna	NULL
5	Kevin	Data
12	NULL	Engineer
35	NULL	Mkt



Join example

```
SELECT A.*, B.*  
FROM customers A  
JOIN invoices B  
ON A.customerid = B.customerid;
```

This query joins two tables -
customers and invoices



Join example

SELECT

A.*,

B.*,

C.*,

D.*

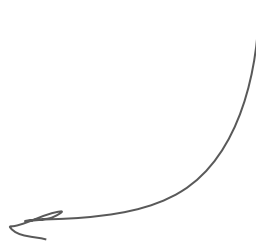
FROM table1 A

JOIN table2 B **ON** A.id = B.id

JOIN table3 C **ON** B.id = C.id

JOIN table4 D **ON** C.id = D.id;

It's easy to join more than two
tables (4-5 tables is quite normal)





Warning! The following slides are **advanced** SQL topics. Be prepared and hope you enjoy.





Repl.it

Repl.it online editor

The screenshot displays the Repl.it online editor interface for a project named "PrimaryVariableSystem". The user "toyeiei" is logged in. The interface includes a file explorer on the left, a central code editor, and a console on the right.

File Explorer: Shows a list of files including "main.sql", "chinook.db", "get_data.sql", "gpa.csv", "main2.sql", and "school.db".

Code Editor: Displays the content of "main.sql", which contains SQL code for creating and inserting data into three tables: students, majors, and locations.

```
1 drop table if exists students;
2 drop table if exists majors;
3 drop table if exists locations;
4
5 create table students (
6   id int primary key,
7   firstname text,
8   lastname text,
9   major int,
10  FOREIGN KEY (major) REFERENCES majors(major_id)
11 );
12
13 create table majors (
14   major_id int primary key,
15   name text
16 );
17
18 create table locations (
19   firstname text,
20   lastname text,
21   city text,
22   PRIMARY KEY (firstname, lastname),
23   FOREIGN KEY (firstname, lastname) REFERENCES students (firstname, lastname)
24 );
25
26 insert into students values
27   (1, 'David', 'Eagle', 1),
28   (2, 'Mary', 'Anne', 2),
29   (3, 'John', 'Travolta', 2);
30
31 insert into majors values
32   (1, 'Economics'),
33   (2, 'Data Science');
34
35 insert into locations values
36   ('David', 'Eagle', 'New York'),
37   ('Mary', 'Anne', 'Moscow'),
38   ('John', 'Keleher', 'London');
```

Console: Shows the output of the SQL execution, displaying "SQLite version 3.22.0" and a prompt character ">".



Essential command lines

Command	What it does?
.help	เรียกดูชื่อ sql commands ทั้งหมด
.open	เปิดไฟล์ database
.read	อ่านไฟล์ sql script
.mode	เปลี่ยน mode การแสดงผล
.header	แสดงชื่อ column ใน terminal
.table	แสดงชื่อ tables ใน database
.schema	แสดง schema ของตารางทั้งหมด
.import	นำเข้า csv file เป็น table



We can also run SQL in terminal

```
Console  Shell
SQLite version 3.22.0
>
> .shell pwd
/home/runner/PrimaryVariableSystem
>
> .shell ls -l
total 908
-rw-r--r-- 1 runner runner 884736 Dec 16 10:20 chinook.db
-rw-r--r-- 1 runner runner    45 Dec 16 10:34 get_data.sql
-rw-r--r-- 1 runner runner    40 Dec 16 11:10 gpa.csv
-rw-r--r-- 1 runner runner   119 Dec 16 11:42 main2.sql
-rw-r--r-- 1 runner runner   851 Dec 16 11:24 main.sql
-rw-r--r-- 1 runner runner 28672 Dec 16 11:24 school.db
>
> .shell cat gpa.csv
id,gpa
1,3.45
2,3.67
3,3.89
4,2.55
>
> .shell echo "Hello world"
Hello world
>
> .open chinook.db
>
> .table
albums      employees  invoices   playlists
artists     genres     media_types tracks
customers   invoice_items playlist_track
>
> select firstname, lastname, country from customers limit 5;
Luis|Gonçalves|Brazil
Leonie|Köhler|Germany
François|Tremblay|Canada
Bjørn|Hansen|Norway
František|Wichterlová|Czech Republic
>
> 
```

← We call this "TERMINAL"



Join syntax

```
SELECT A.*, B.*  
FROM customers A  
JOIN invoices B  
ON A.customerid = B.customerid;
```

Alias or shorter name

PK = FK



Join using

```
SELECT A.*, B.*  
FROM customers A  
JOIN invoices B  
USING (customerid);
```



Using if the column names in
both tables are the same



Join more than one column

```
SELECT A.*, B.*  
FROM tableA A  
JOIN tableB B  
    ON A.customerid = B.customerid  
    AND A.country = B.country;
```

 use AND to add more columns to join



Review CASE syntax

```
SELECT  
  CASE  
    WHEN company IS NULL THEN 'End Customers'  
    ELSE 'Corporate'  
  END AS segment  
FROM customers;
```

Condition

Value if True

Value if False

The diagram shows three handwritten annotations with arrows pointing to specific parts of the SQL code. The annotation 'Condition' has an arrow pointing to the 'company IS NULL' part of the WHEN clause. The annotation 'Value if True' has an arrow pointing to the 'End Customers' string in the THEN clause. The annotation 'Value if False' has an arrow pointing to the 'Corporate' string in the ELSE clause.



Case + Aggregate Functions

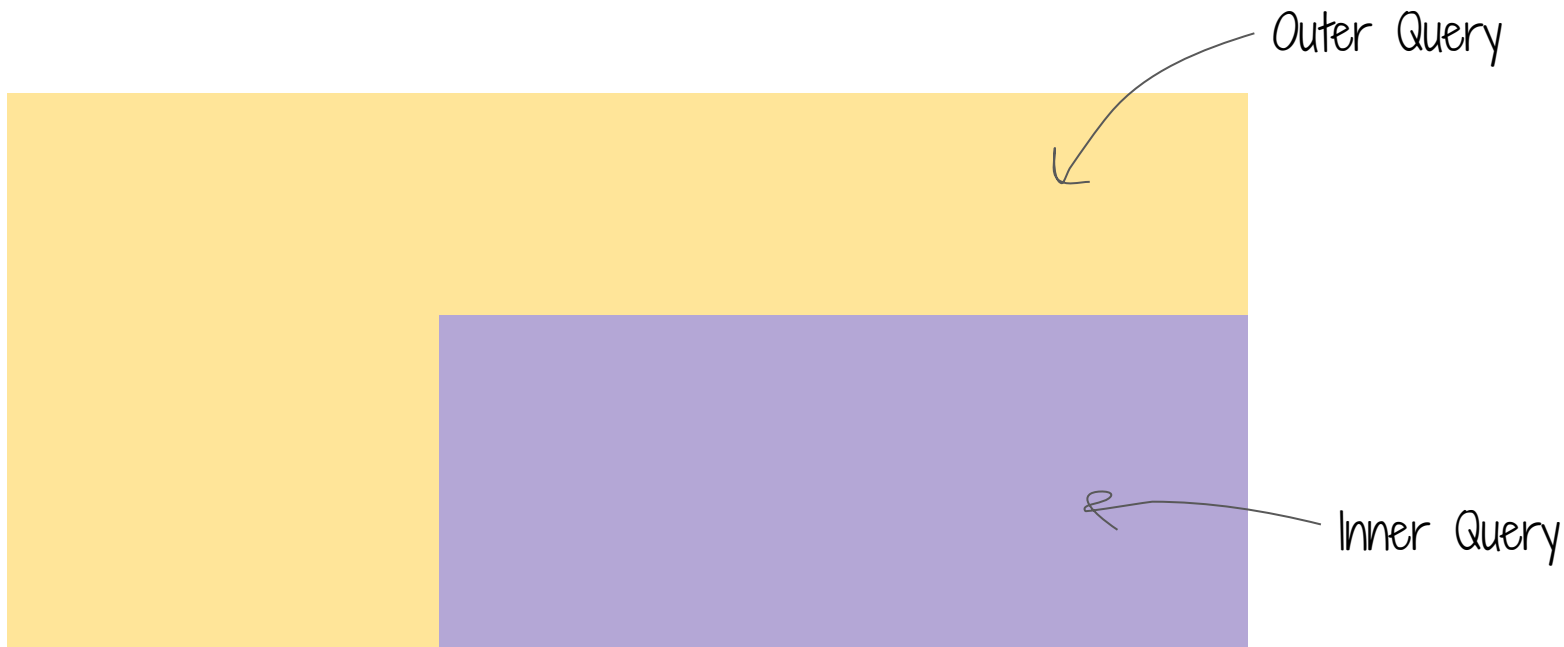
```
SELECT  
  CASE  
    WHEN company IS NULL THEN 'End Customers'  
    ELSE 'Corporate'  
  END AS segment,  
  COUNT(*) AS N  
FROM customers  
GROUP BY 1;
```

Count customers in each segment





Intro to subqueries





Inner run first

```
SELECT firstname, lastname, country FROM (  
    SELECT * FROM customers  
    WHERE country IN ('USA', 'United Kingdom', 'Canada')  
)  
ORDER BY 3 DESC;
```

↑
Inner Query



Outer run later

```
SELECT firstname, lastname, country FROM (  
  
    SELECT * FROM customers  
    WHERE country IN ('USA', 'United Kingdom', 'Canada')  
)  
  
ORDER BY 3 DESC;
```



Outer Query



We often use subqueries in where clause

```
SELECT * FROM tracks  
WHERE bytes = (  
    SELECT MAX(bytes) FROM tracks  
);
```

Find max bytes



Intro to window functions

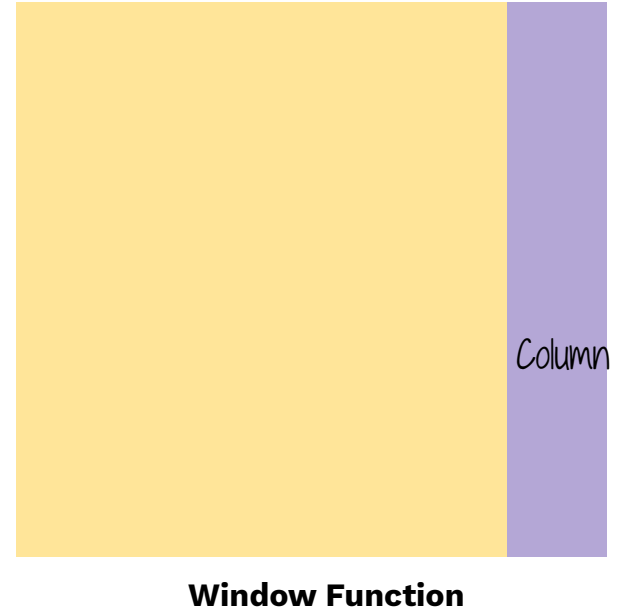
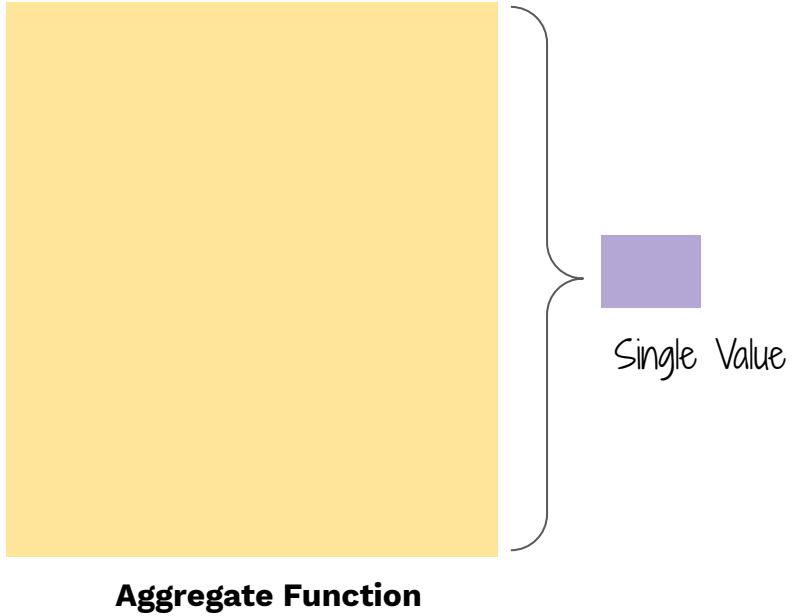


Two things you need to know about window functions

1. they are in **SELECT** clause
2. they create new columns in the result table



Aggregate vs. Window functions





Window function syntax

```
WINDOW_FUNCTION( ) OVER( PARTITION BY ... ORDER BY ... )
```

Function Name

Create window



The easiest window function

```
SELECT
    firstname,
    lastname,
    ROW_NUMBER() OVER() AS rowNum
FROM customers;
```

FirstName	LastName	rowNum
Luís	Gonçalves	1
Leonie	Köhler	2
François	Tremblay	3
Bjørn	Hansen	4
František	Wichterlová	5
Helena	Holý	6
Astrid	Gruber	7
Daan	Peeters	8
Kara	Nielsen	9
Eduardo	Martins	10
Alexandre	Rocha	11
Roberto	Almeida	12
Fernanda	Ramos	13
Mark	Philips	14
Jennifer	Peterson	15



Common window functions

Function	What it does?
ROW_NUMBER()	สร้างคอลัมน์ row number เรียงตั้งแต่ 1 - n
RANK()	สร้างคอลัมน์ ranking
DENSE_RANK()	สร้างคอลัมน์ ranking
LAG()	สร้างคอลัมน์ LAG value (t-1)
LEAD()	สร้างคอลัมน์ LEAD value (t+1)
NTILE()	สร้างคอลัมน์ segment จัดกลุ่ม records
SUM() OVER()	สร้างคอลัมน์ผลรวมแบบ running total

Bootcamp Live 02 Intro to Databases with



Website: <https://datarockie.com>

