Data Science Workshop

Safouene Kallel

2022-11-16

Setup

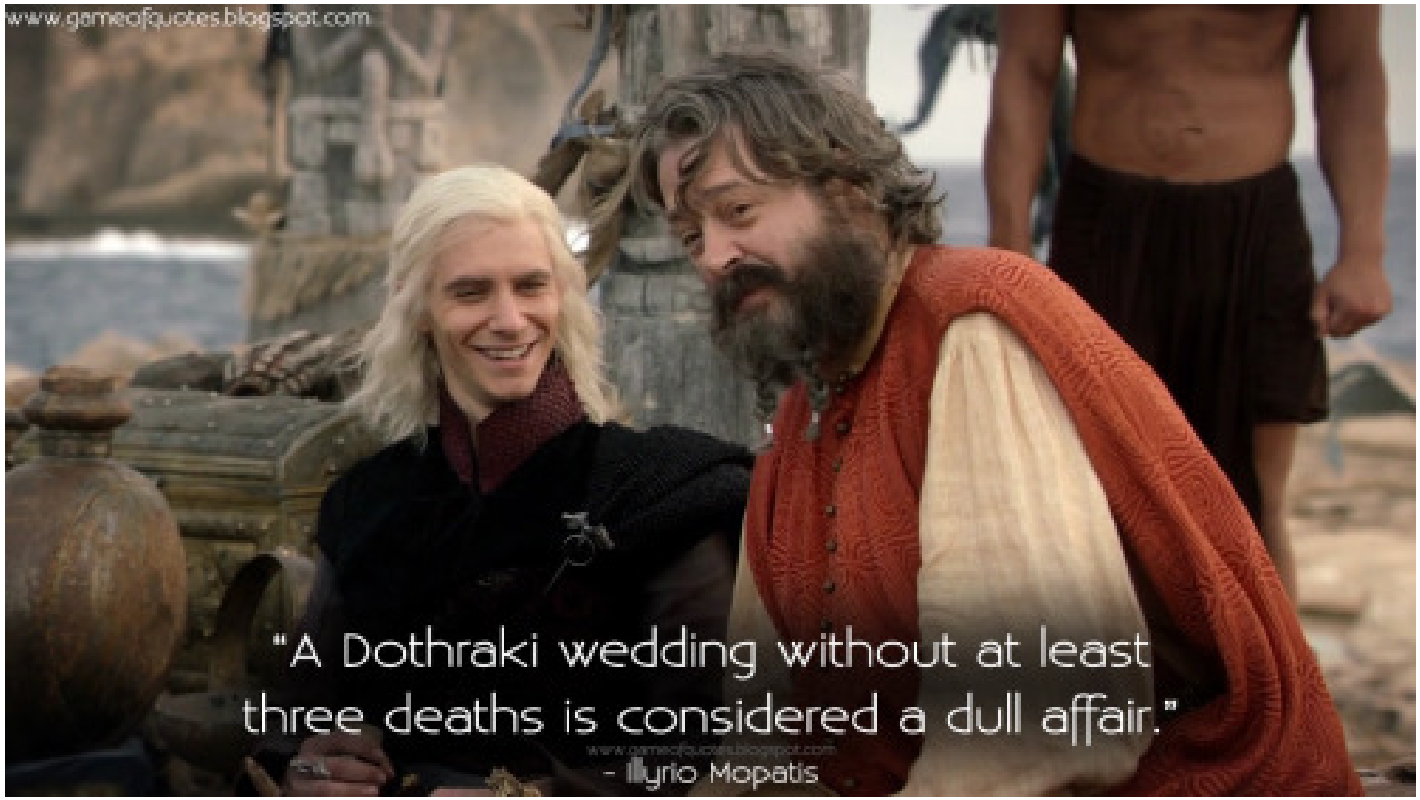

Figure 1: Dothraki Weddings

Setup

Figure 2: We finished the Setup

Session 1

1. Introduction
2. SQL Queries
3. Database Model

What is SQL

SQL: **Structured** Query Language

is a **declarative** language.

"a style of building the structure and elements of computer programs—that expresses the logic of a computation **without describing its control flow**."
opposite to "**imperative** programming [...] that uses statements that change a program's state.

SQL Connection with the MySQL database

Please Connect to the Server:

- server : “stgux-db.vih.infineon.com”
- username : “DS_Training_u01”
- database : “DS_Training”
- port : 3306
- password : “ ”

using mySQL WorkBench.

Please look inside the content of the DS_Training Database.

Get started with SQL Queries

- Right Click on the Customers Table Select 1000 first Rows
- Please run the Query
- Congrats you made your first Query!
- "You Selected the first 1000 rows" **Declarative**
- By default the tool will specify all the columns

SELECT

```
SELECT * FROM DS_Training.customers
```

```
SELECT DISTINCT FirstName FROM DS_Training.employees
```

```
SELECT DISTINCT FirstName, LastName FROM DS_Training.employees
```

WHERE

```
SELECT * FROM DS_Training.employees
WHERE EmployeeID = 2
```

```
SELECT LastName FROM DS_Training.employees
WHERE EmployeeID = 2
```

Possible Operators in the WHERE Clause:

```
> < >= <= = <> BETWEEN LIKE IN
```

AND OR NOT

```
SELECT * FROM DS_Training.Sales
WHERE Quantity > 500 AND SalesPersonID = 2
```

```
SELECT * FROM DS_Training.Sales
WHERE Quantity > 500 OR SalesPersonID = 2
```

```
SELECT * FROM DS_Training.Sales
WHERE Quantity > 500 AND SalesPersonID = 2 OR SalesPersonID =
```

ORDER BY

```
SELECT * FROM DS_Training.Sales
ORDER BY Quantity
```

```
SELECT * FROM DS_Training.Sales
ORDER BY Quantity DESC
```

```
SELECT * FROM DS_Training.Sales
ORDER BY Quantity DESC, SalesPersonID ASC
```

SQL AGGREGATE

The SQL Server is capable of running statistics over the data and giving back the results. So instead of:

- downloading the data
- saving it on your disk
- loading it in memory with another software
- running Statistics

You can request the SQL server to run the statistics and give back the result directly. (It is really handy if the data is big). In our Sales Table we have over 6M rows. It is nearly impossible to load the table in Excel.

MIN, MAX

```
SELECT MAX(Quantity)
FROM DS_Training.Sales
```

```
SELECT MIN(Quantity) AS SmallestQuantity
FROM DS_Training.Sales
```

AVG, SUM, COUNT

```
SELECT
AVG(Quantity) AS AverageQuantity,
SUM(Quantity) AS OverallSales,
COUNT(SalesID) AS SalesNumber
FROM DS_Training.Sales
WHERE ProductID = 5
```

GROUP BY

```
SELECT AVG(Quantity) AS AverageQuantity,
       SUM(Quantity) AS OverallSales,
       COUNT(SalesID) AS SalesNumber
FROM DS_Training.Sales
GROUP BY ProductID
ORDER BY OverallSales
```

HAVING

The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions

```
SELECT SalesPersonID, COUNT(SalesID) AS N
FROM DS_Training.Sales
GROUP BY SalesPersonID
HAVING COUNT(SalesID) > 30000
ORDER BY N
```

SQL is Relational

We have been working on tables independently for the past minutes. Now we discover the true superpower of SQL.

SQL PRIMARY KEY

The PRIMARY KEY uniquely identifies each record in a table. Primary Keys must contain UNIQUE values. A Table can have only ONE primary key. The PRIMARY KEY can consist of multiple columns.

SQL FOREIGN KEY

The FOREIGN KEY is a field in one table that refers to the PRIMARY Key in another table.

The table with the foreign key is called the child table, and the table with the primary key is called the referenced or parent table.

Database Model

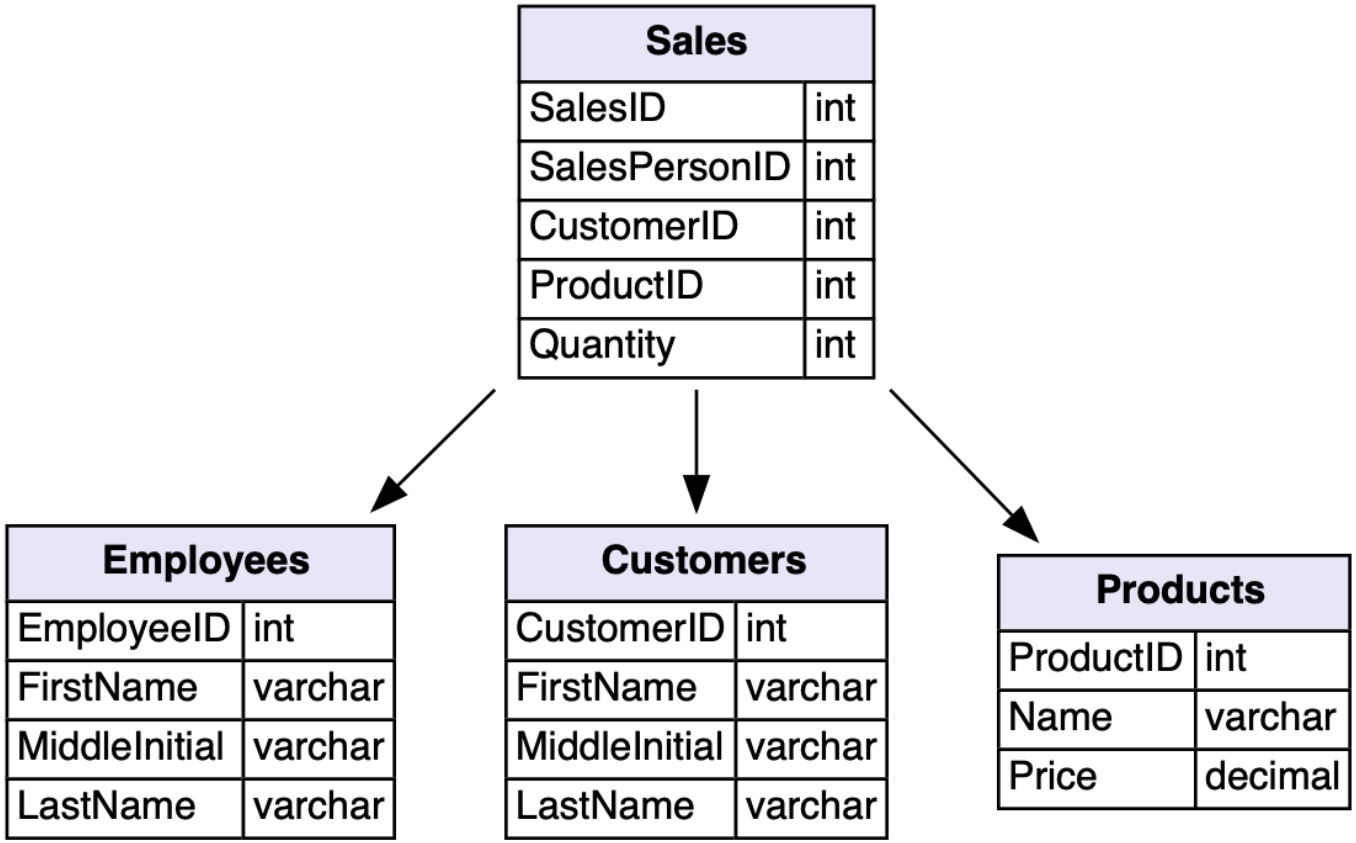

Figure 3: Database Model of SalesDB

INNER JOIN

```
SELECT DS_Training.Sales.*, DS_Training.customers.*
FROM DS_Training.Sales
INNER JOIN DS_Training.customers
ON DS_Training.customers.CustomerID = DS_Training.Sales.Custom
```

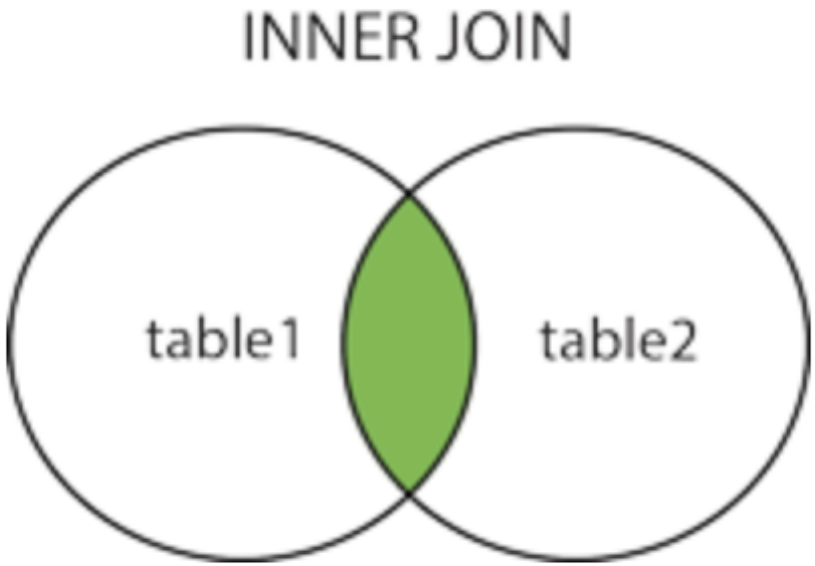

Figure 4: JOING TABLES

LEFT JOIN

```
SELECT DS_Training.Sales.*, DS_Training.customers.*
FROM DS_Training.Sales
LEFT JOIN DS_Training.customers
ON DS_Training.customers.CustomerID = DS_Training.Sales.Custom
```

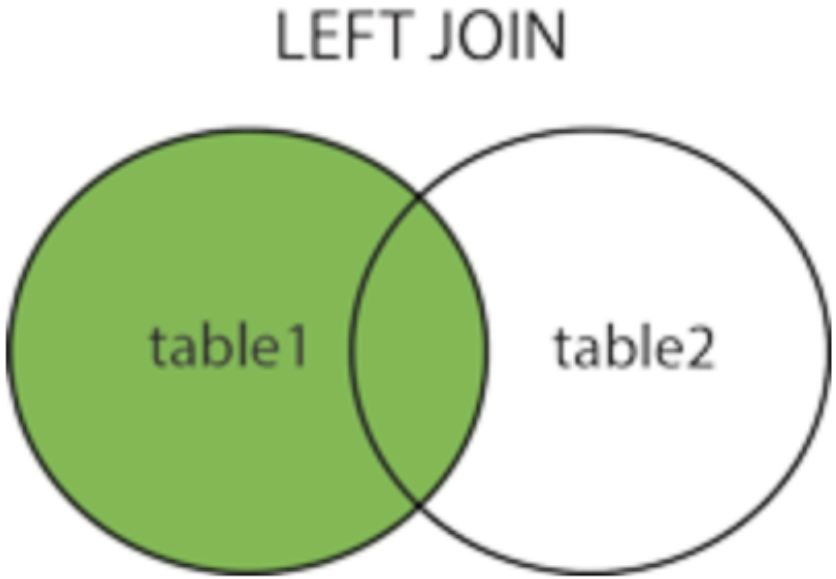

Figure 5: JOING TABLES

FULL JOIN

```
SELECT DS_Training.Sales.*, DS_Training.customers.*
FROM DS_Training.Sales
FULL JOIN DS_Training.customers
ON DS_Training.customers.CustomerID = DS_Training.Sales.Custom
```

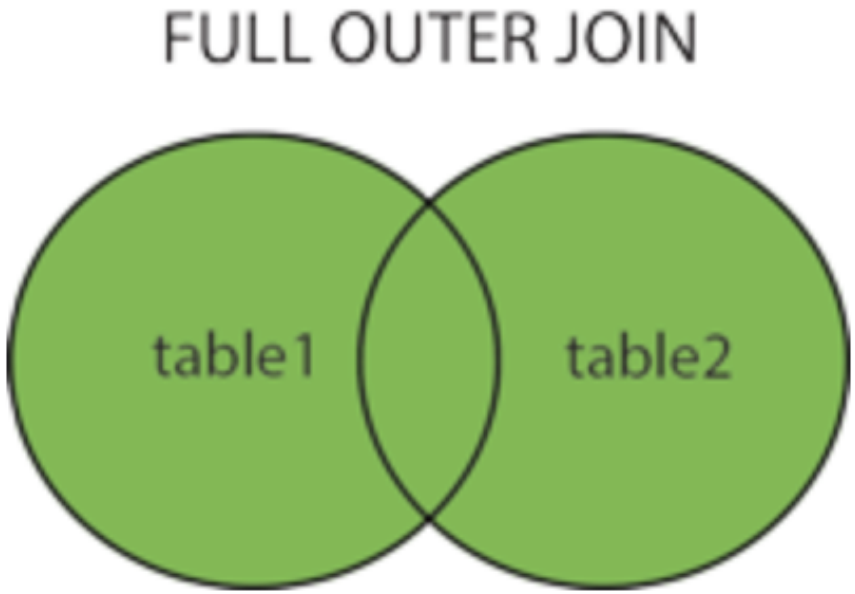

Figure 6: JOING TABLES

ACID

- ATOMICITY (Money transfer)
- CONSISTENCY (Data is valid according to all defined rules)
- ISOLATION (Bank concurrent transactions)
- DURABILITY (None Volatile Memory)

Assignment

Try to re-do this on your own, using the Training Database or another database that you might have access to.