COMS 4111: **Introduction to Databases**

Lecture 4: Data Modeling, Relationships, Constraints, JOIN, HW 2

Dr. Donald F. Ferguson
dff9@columbia.edu

Contents

Contents

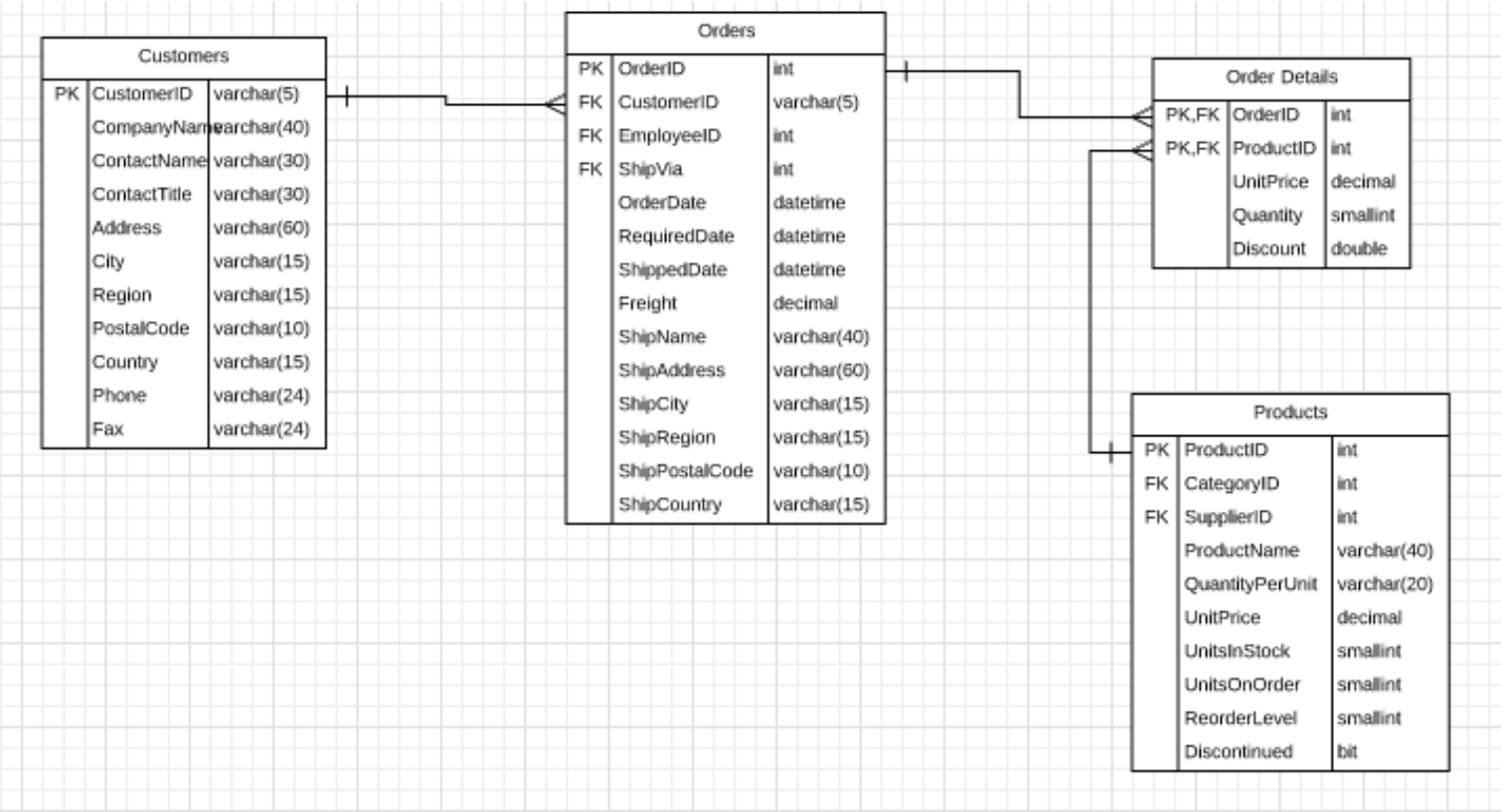
- Introduction: Questions? Comments?
- Data Modeling (II) – Next Level
  - Relationships.
  - Entity Classifications.
  - Introduction to Constraints and Foreign Keys
- SQL
  - JOIN
  - Scenarios and SQL
- Assignment 2.

Introduction

Questions?
Comments?

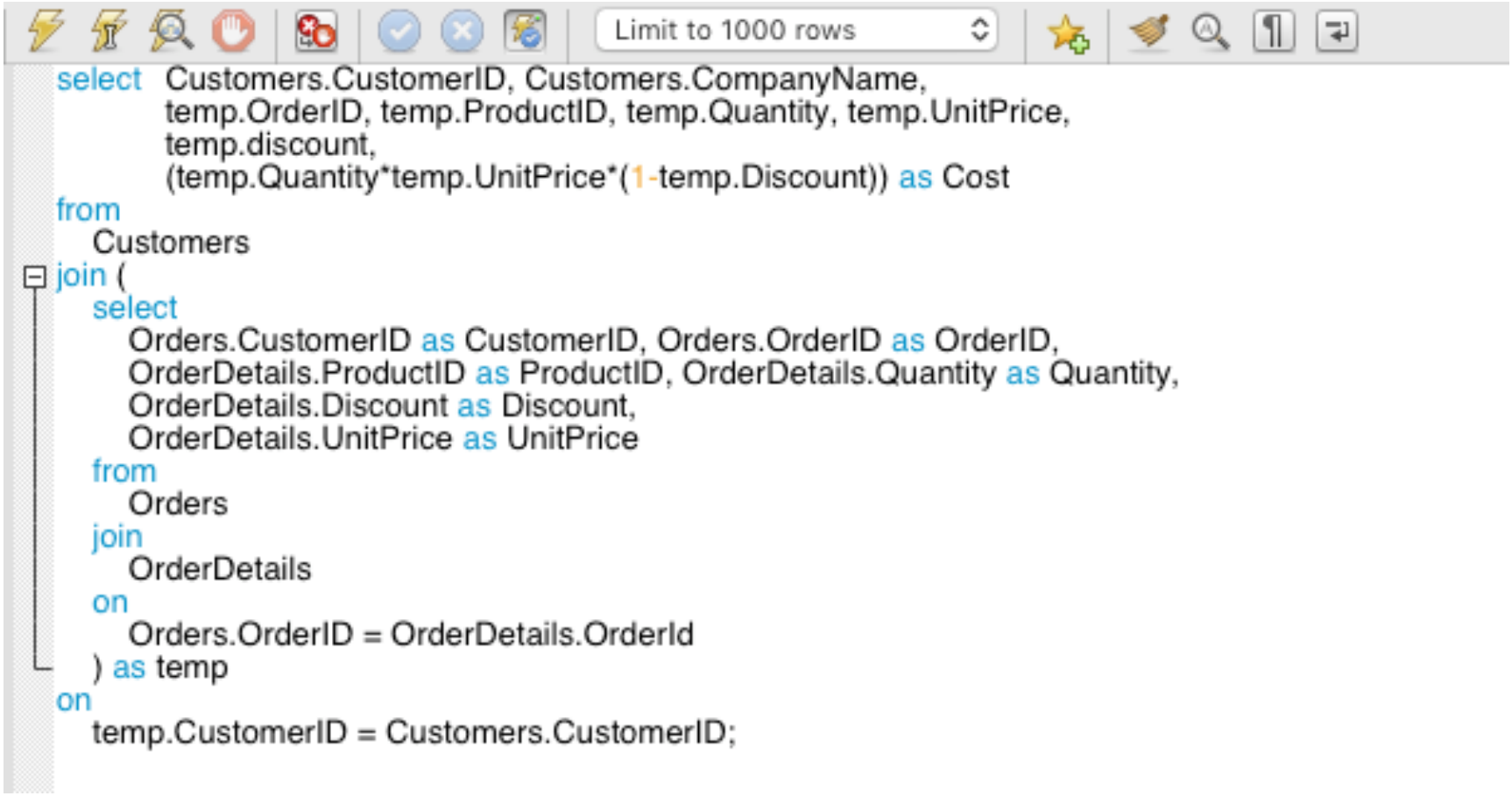
Mind Officially Blown

Data Model and a Question

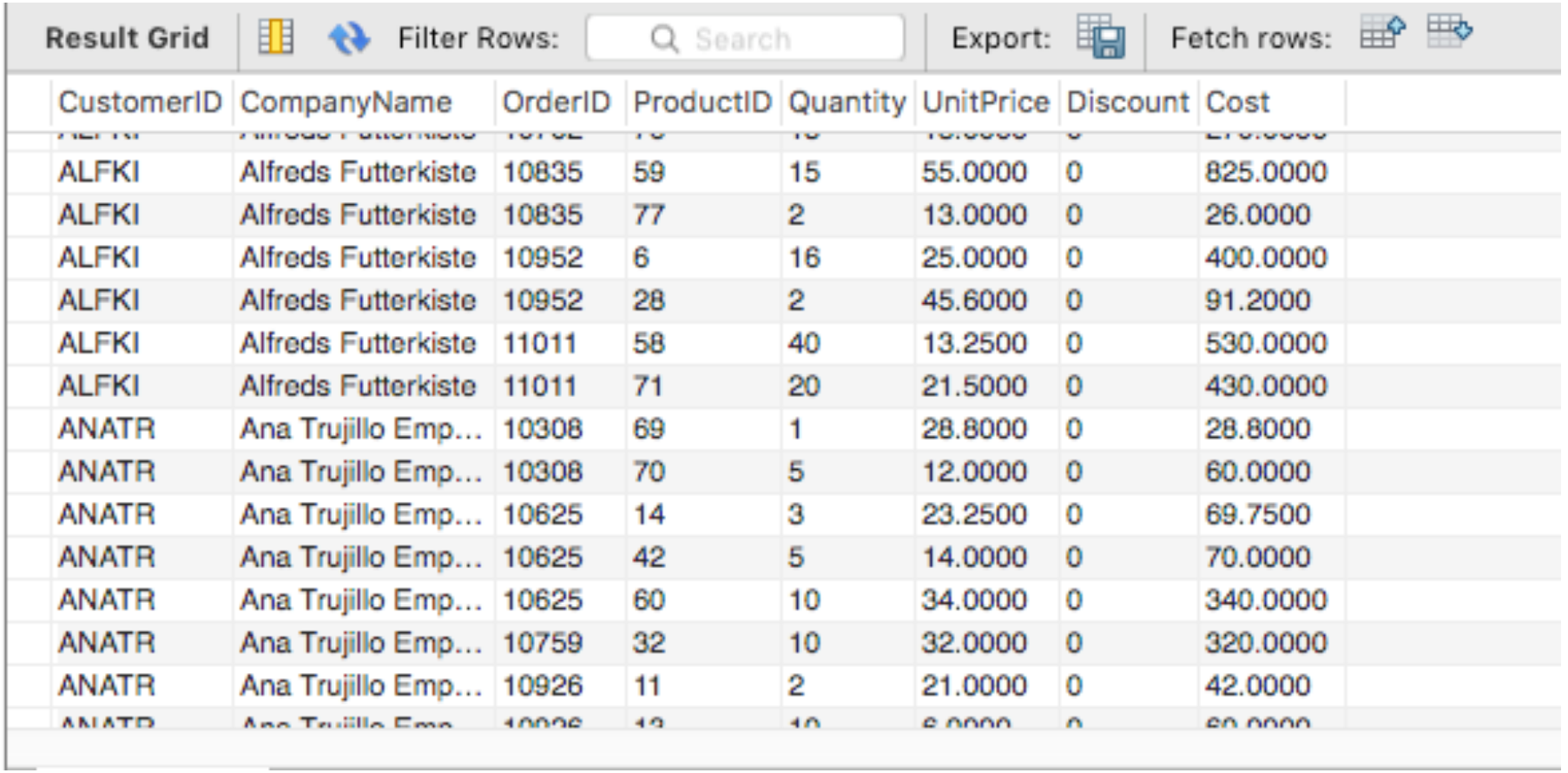


I want to know every CompanyName, ProductID, Amount combination.

Mind Blown



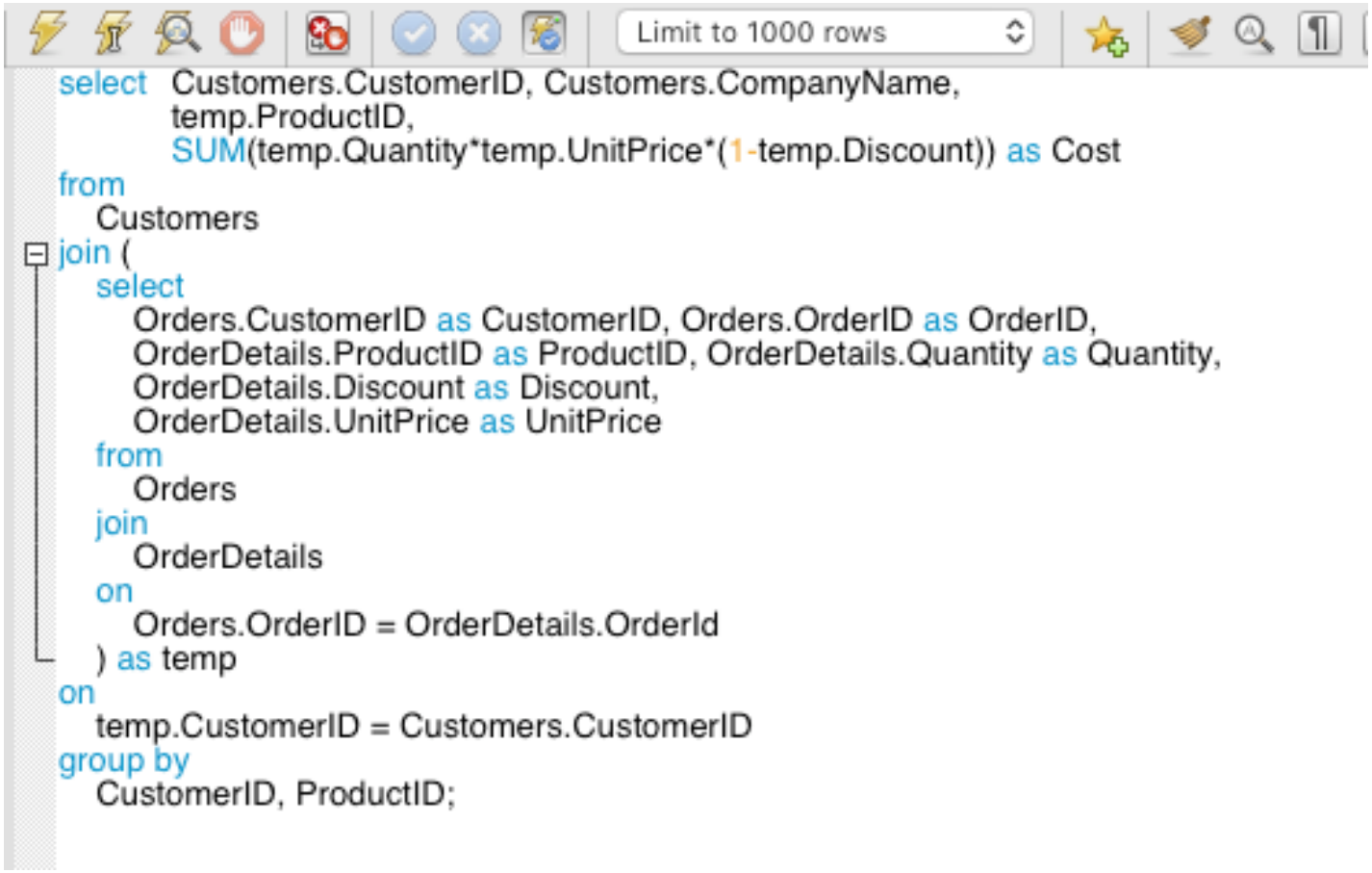
Mind Blown



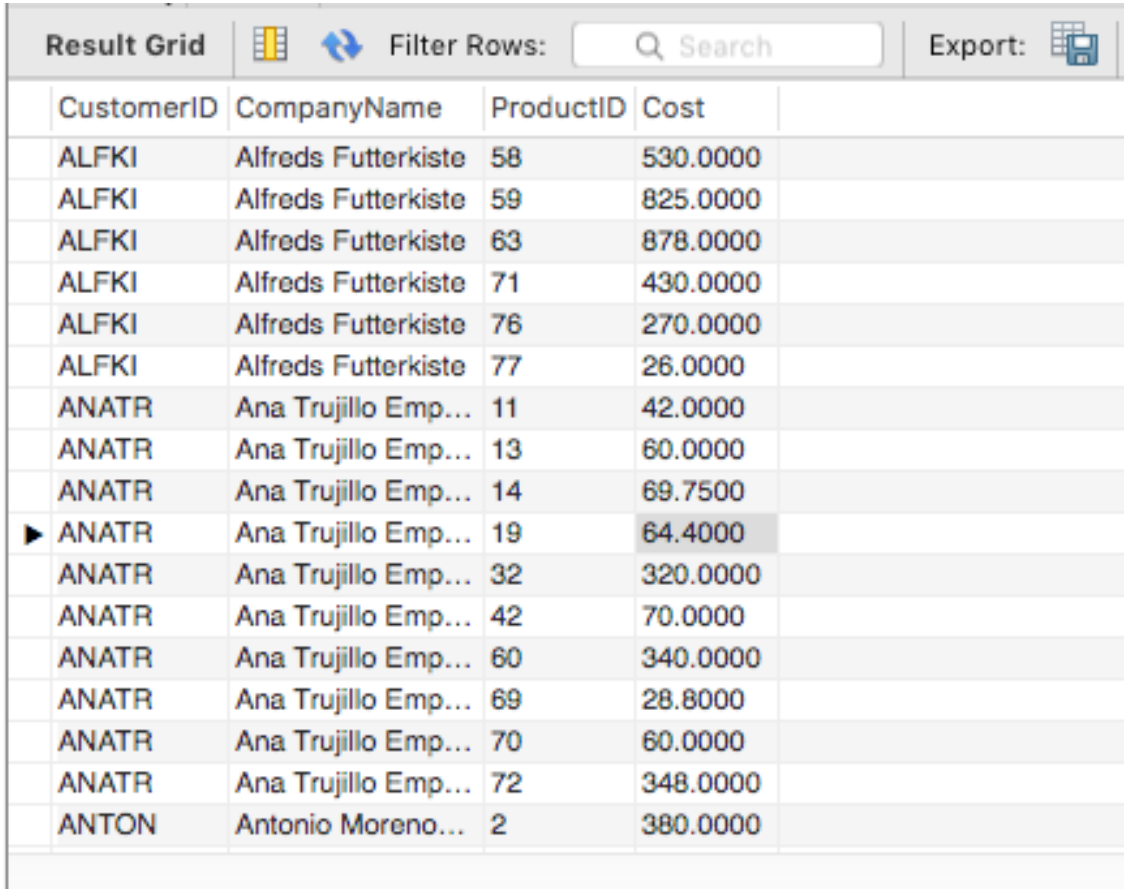
Result Grid | Filter Rows: Search | Export: | Fetch rows:

CustomerID	CompanyName	OrderID	ProductID	Quantity	UnitPrice	Discount	Cost
ALFKI	Alfreds Futterkiste	10835	59	15	55.0000	0	825.0000
ALFKI	Alfreds Futterkiste	10835	77	2	13.0000	0	26.0000
ALFKI	Alfreds Futterkiste	10952	6	16	25.0000	0	400.0000
ALFKI	Alfreds Futterkiste	10952	28	2	45.6000	0	91.2000
ALFKI	Alfreds Futterkiste	11011	58	40	13.2500	0	530.0000
ALFKI	Alfreds Futterkiste	11011	71	20	21.5000	0	430.0000
ANATR	Ana Trujillo Emp…	10308	69	1	28.8000	0	28.8000
ANATR	Ana Trujillo Emp…	10308	70	5	12.0000	0	60.0000
ANATR	Ana Trujillo Emp…	10625	14	3	23.2500	0	69.7500
ANATR	Ana Trujillo Emp…	10625	42	5	14.0000	0	70.0000
ANATR	Ana Trujillo Emp…	10625	60	10	34.0000	0	340.0000
ANATR	Ana Trujillo Emp…	10759	32	10	32.0000	0	320.0000
ANATR	Ana Trujillo Emp…	10926	11	2	21.0000	0	42.0000

Mind Blown



Mind Blown



Result Grid | Filter Rows: Search | Export:

CustomerID	CompanyName	ProductID	Cost
ALFKI	Alfreds Futterkiste	58	530.0000
ALFKI	Alfreds Futterkiste	59	825.0000
ALFKI	Alfreds Futterkiste	63	878.0000
ALFKI	Alfreds Futterkiste	71	430.0000
ALFKI	Alfreds Futterkiste	76	270.0000
ALFKI	Alfreds Futterkiste	77	26.0000
ANATR	Ana Trujillo Emp…	11	42.0000
ANATR	Ana Trujillo Emp…	13	60.0000
ANATR	Ana Trujillo Emp…	14	69.7500
ANATR	Ana Trujillo Emp…	19	64.4000
ANATR	Ana Trujillo Emp…	32	320.0000
ANATR	Ana Trujillo Emp…	42	70.0000
ANATR	Ana Trujillo Emp…	60	340.0000
ANATR	Ana Trujillo Emp…	69	28.8000
ANATR	Ana Trujillo Emp…	70	60.0000
ANATR	Ana Trujillo Emp…	72	348.0000
ANTON	Antonio Moreno…	2	380.0000

SELECT … WHERE … GROUP BY …

Question about "SELECT Statement with Group By"

I try the example "SELECT Country,count(*) FROM northwind.Customers group by Country ORDER by Country;" and the result is:

Country	count(*)
Argentina	3
Austria	2
Belgium	2
Brazil	9
Canada	3

SELECT Country, count(*) FROM northwind.Customers group by Country ORDER by Country;

And then I tried "SELECT Country,CustomerID FROM northwind.Customers ORDER by Country;" and find that three customers from Argentina have different CustomerID.

Country	CustomerID
Argentina	CACTU
Argentina	RANCH
Argentina	OCEAN
Austria	PICCO
Austria	ERNSH
Belgium	SUPRD
Belgium	MAISD
Brazil	FAMIA

SELECT Country, CustomerID FROM northwind.Customers ORDER by Country;"

SELECT … WHERE … GROUP BY …

Then I tried: "SELECT Country,count(*),CustomerID FROM northwind.Customers group by Country ORDER by Country;" And find that it only shows one CustomerID.

Country	count(*)	CustomerID
Argentina	3	CACTU
Austria	2	ERNSH
Belgium	2	MAISD
Brazil	9	COMMI
Canada	3	BOTTM
Denmark	2	SIMOB

```
SELECT Country, count(*), CustomerID
FROM northwind.Customers
group by Country ORDER by Country;
```

My question is what's the reason of this result, because I hope the result can be Argentina, 3, (C
And if I want the answer to be Argentina, 3, (CACTU, RANCH, OCEAN), how should I modify the

What the student wanted was

Country	noOfCustomers	Customers
Argentina	3	(OCEAN,RANCH,CACTU)
Austria	2	(ERNSH,PICCO)
Belgium	2	(SUPRD,MAISD)
Brazil	9	(QUEEN,QUEDE,COMMI,WELLI,FAMIA,TRAD…
Canada	3	(BOTTM,LAUGB,MEREP)
Denmark	2	(VAFFE,SIMOB)
Finland	2	(WARTH,WILMK)
France	11	(BLONP,VINET,VICTE,BONAP,DUMON,FOLIG,…
Germany	11	(TOMSP,KOENE,FRANK,QUICK,LEHMS,DRA…
Ireland	1	(HUNGO)
Italy	3	(REGGC,FRANS,MAGAA)
Mexico	5	(TORTU,CENTC,PERIC,ANTON,ANATR)
Norway	1	(SANTG)
Poland	1	(WOLZA)

Observation 1

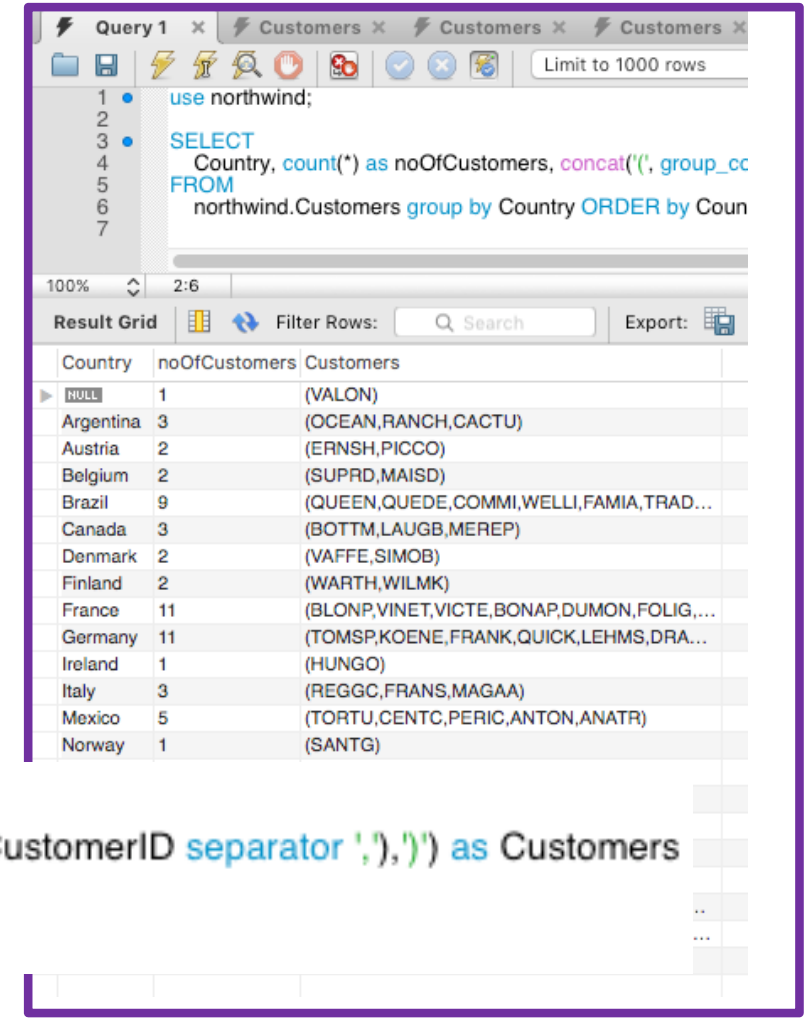
- What is the domain (type) of Customers?
  - Domains must be "atomic."
  - Some examples
    - String
    - Number
    - Date or Time
- What is the intent of (c1, c2, c3, c4)?
  - Array inside a column?
  - Tuple inside a column in a tuple?
  - Struct?

None of these compound types are validin the relational model.

Country	noOfCustomers	Customers
Argentina	3	(OCEAN,RANCH,CACTU)
Austria	2	(ERNSH,PICCO)
Belgium	2	(SUPRD,MAISD)
Brazil	9	(QUEEN,QUEDE,COMMI,WELLI,FAMIA,TRAD...
Canada	3	(BOTTM,LAUGB,MEREP)
Denmark	2	(VAFFE,SIMOB)
Finland	2	(WARTH,WILMK)
France	11	(BLONP,VINET,VICTE,BONAP,DUMON,FOLIG...
Germany	11	(TOMSP,KOENE,FRANK,QUICK,LEHMS,DRA...
Ireland	1	(HUNGO)
Italy	3	(REGGC,FRANS,MAGAA)
Mexico	5	(TORTU,CENTC,PERIC,ANTON,ANATR)
Norway	1	(SANTG)
Poland	1	(WOLZA)
Portugal	2	(PRINI,FURIB)
Spain	5	(BOLID,FISSA,ROMEY,GALED,GODOS)
Sweden	2	(BERGS,FOLKO)
Switzerl...	2	(CHOPS,RICSU)

Continued

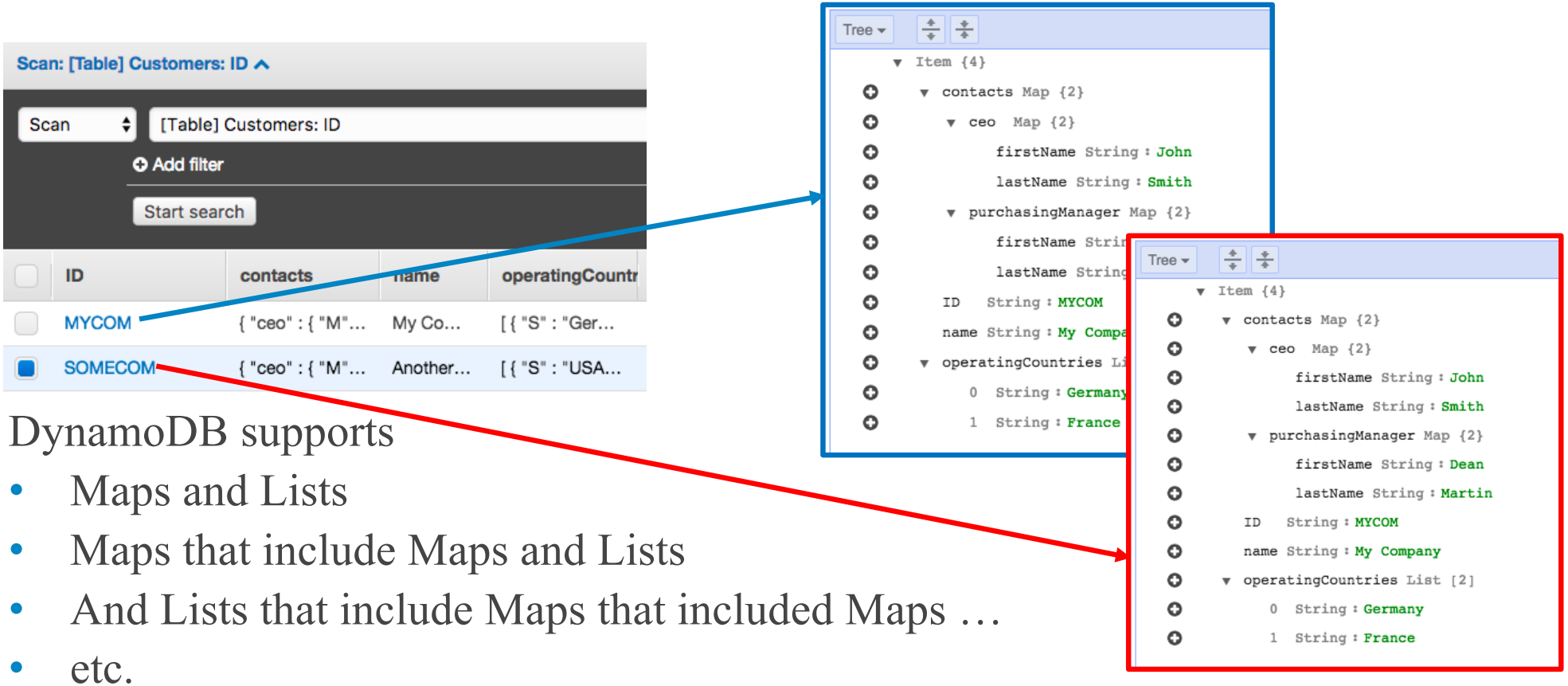
- Well Don, you clearly did this somehow.
- There are two valid types of arrays
  - String is an array of characters.
  - A *blob* is an array of bytes used to hold an unstructured "chunk" of data, e.g.
    - Image
    - Zip file
- I used a string.




```
SELECT
   Country, count(*) as noOfCustomers, concat('(', group_concat(CustomerID separator ','),')') as Customers
FROM
   northwind.Customers group by Country ORDER by Country;
```

Multi-Valued Attributes

Some Databases Support More Complex Attributes Types, e.g. DynamoDB

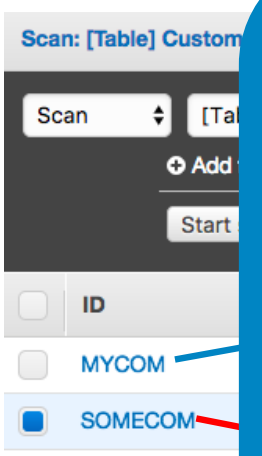


DynamoDB supports

- Maps and Lists
- Maps that include Maps and Lists
- And Lists that include Maps that included Maps …
- etc.

Multi-Valued Attributes

Some Databases Support More Complex Attributes Types, e.g. DynamoDB



We will cover some alternative database models, but in less detail than relational. Most likely

- DynamoDB
- Neo4J
- Redis

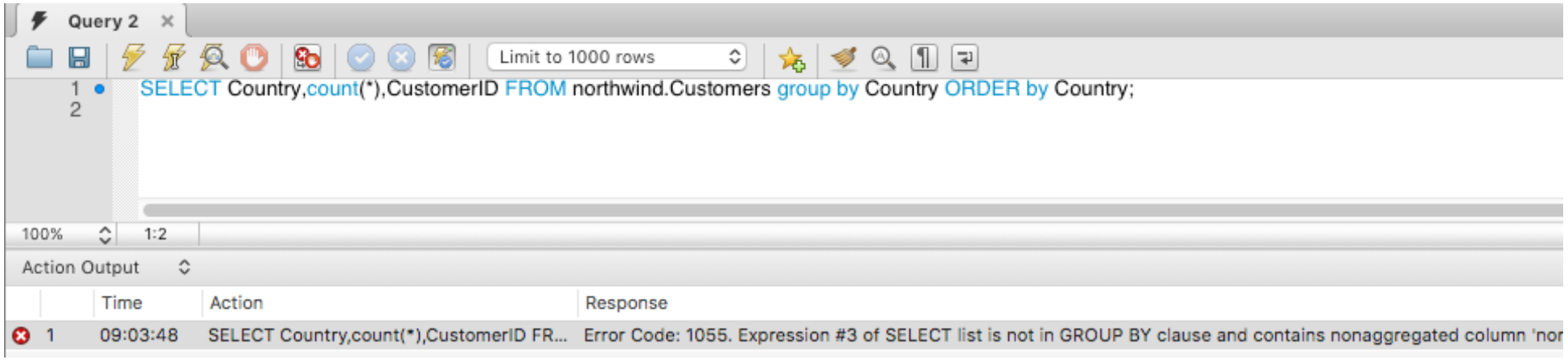
Including

- Motivation, Pros/Cons
- Scenarios

DynamoD

- Maps and
- Including Maps that include Maps and Lists
- And Lists that include Maps that included Maps …

What Happens when I Run the Desired Query?

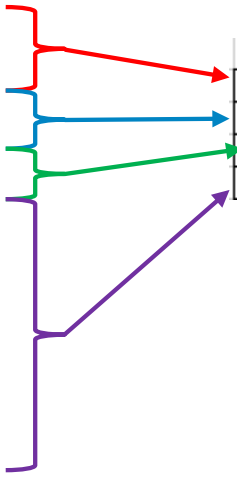


Error Code: 1055. Expression #3 of SELECT list is not in GROUP BY clause and contains nonaggregated column 'northwind.Customers.CustomerID' which is not functionally dependent on columns in GROUP BY clause; this is incompatible with sql_mode=only_full_group_by

GROUP BY

Maps a Set (Group) of Tuples into a Single Tuple

Country	CompanyID	CompanyName	ContactName	ContactTitle	Address	City	Region
Argentina	OCEAN	Ocano Atlntico Ltda.	Yvonne Moncada	Sales Agent	Ing. Gustavo	Buenos Aires	
Argentina	RANCH	Rancho grande	Sergio Gutirrez	Sales Representative	Av. del Liber	Buenos Aires	
Argentina	CACTU	Cactus Comidas para lleva	Patricio Simpson	Sales Agent	Cerrito 333	Buenos Aires	
Austria	ERNSH	Ernst Handel	Roland Mendel	Sales Manager	Kirchgasse 6	Graz	
Austria	PICCO	Piccolo und mehr	Georg Pipps	Sales Manager	Geislweg 14	Salzburg	
Belgium	SUPRD	Suprmes dlices	Pascale Cartrain	Accounting Manager	Boulevard Ti	Charleroi	
Belgium	MAISD	Maison Dewey	Catherine Dewey	Sales Agent	Rue Joseph-I	Bruxelles	
Brazil	QUEEN	Queen Cozinha	Lcia Carvalho	Marketing Assistant	Alameda dos	Sao Paulo	SP
Brazil	QUEDE	Que Delcia	Bernardo Batista	Accounting Manager	Rua da Panif	Rio de Janeir	RJ
Brazil	COMMI	Comrcio Mineiro	Pedro Afonso	Sales Associate	Av. dos Lusa	Sao Paulo	SP
Brazil	WELLI	Wellington Importadora	Paula Parente	Sales Manager	Rua do Merc	Resende	SP
Brazil	FAMIA	Familia Arquibaldo	Aria Cruz	Marketing Assistant	Rua Ors, 92	Sao Paulo	SP
Brazil	TRADH	Tradio Hipermercados	Anabela Domingues	Sales Representative	Av. Ins de Ca	Sao Paulo	SP
Brazil	GOURL	Gourmet Lanchonetes	Andr Fonseca	Sales Associate	Av. Brasil, 44	Campinas	SP
Brazil	RICAR	Ricardo Adocicados	Janete Limeira	Assistant Sales Agent	Av. Copacab	Rio de Janeir	RJ
Brazil	HANAR	Hanari Carnes	Mario Pontes	Accounting Manager	Rua do Pao,	Rio de Janeir	RJ



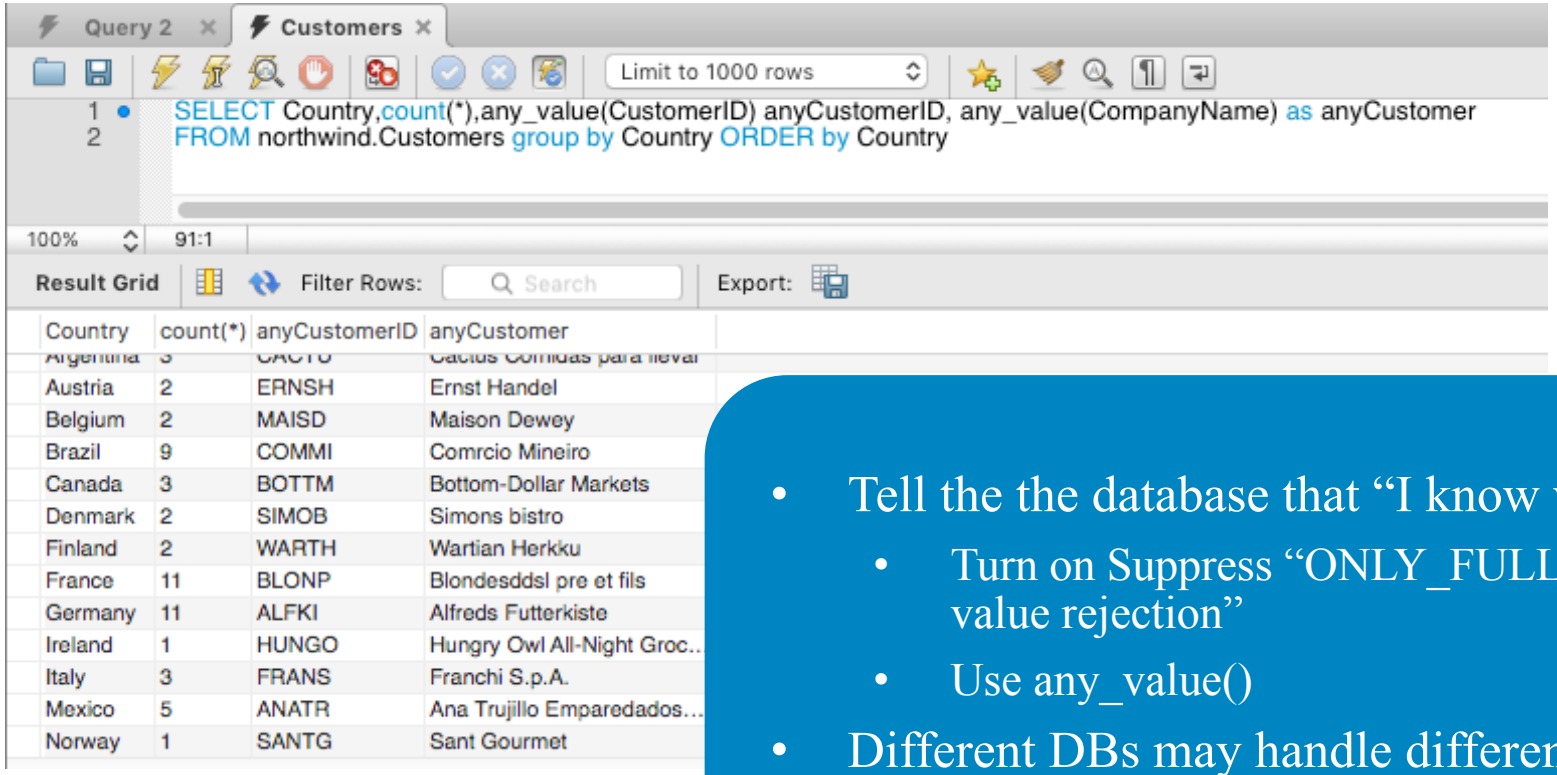
Country	C1	C2	C3
Argentina	V11	V21	V31
Austria	V21	V22	V23
Belgium	V31	V32	V33
Brazil	V41	V42	V43

- This means that V11, V12 and V13 must be some Function
- That maps

Ocano Atlntico Ltda.	Yvonne Moncada	Sales Agent	Ing. Gustavo	Buenos Aires
Rancho grande	Sergio Gutirrez	Sales Representative	Av. del Liber	Buenos Aires
Cactus Comidas para lleva	Patricio Simpson	Sales Agent	Cerrito 333	Buenos Aires

- into a valid relational attribute (domain) type.

A Query that Works …



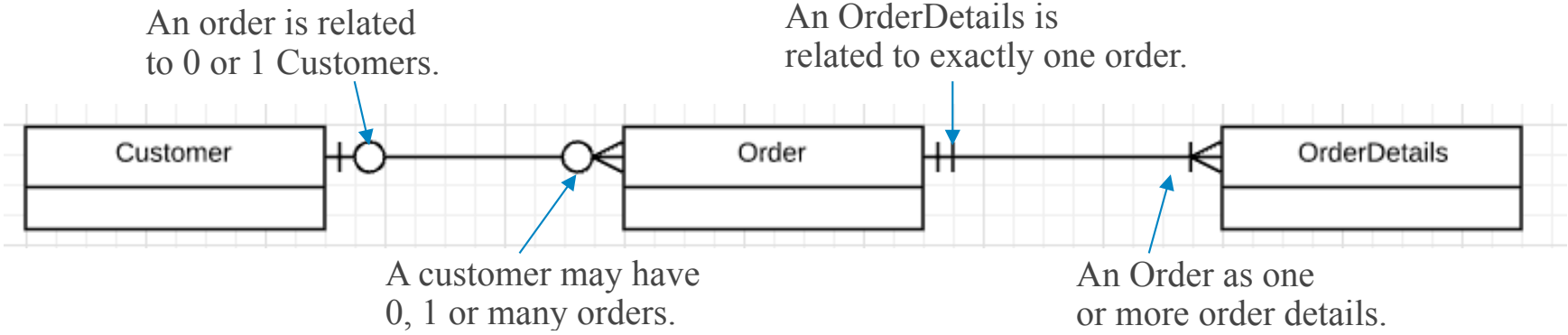
Country	count(*)	anyCustomerID	anyCustomer
Austria	2	ERNSH	Ernst Handel
Belgium	2	MAISD	Maison Dewey
Brazil	9	COMMI	Comrcio Mineiro
Canada	3	BOTTM	Bottom-Dollar Markets
Denmark	2	SIMOB	Simons bistro
Finland	2	WARTH	Wartian Herkku
France	11	BLONP	Blondesddsl pre et fils
Germany	11	ALFKI	Alfreds Futterkiste
Ireland	1	HUNGO	Hungry Owl All-Night Groc..
Italy	3	FRANS	Franchi S.p.A.
Mexico	5	ANATR	Ana Trujillo Emparedados…
Norway	1	SANTG	Sant Gourmet

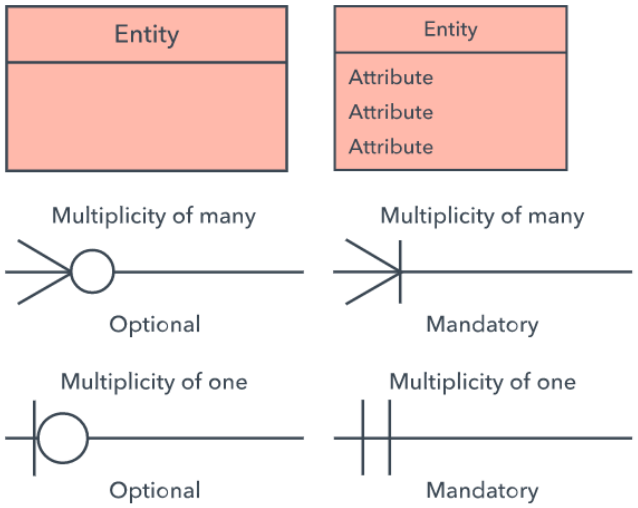
- Tell the the database that "I know what I doing!"
  - Turn on Suppress "ONLY_FULL_GROUP_BY value rejection"
  - Use any_value()
- Different DBs may handle differently.

Data Modeling

More Detail

Northwind Relationships

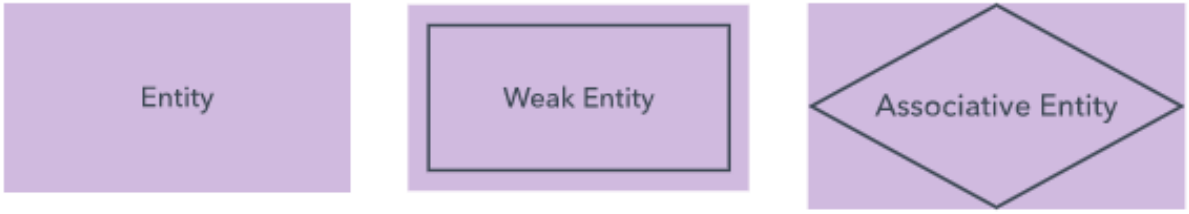




- This *notation* is called
  - Crow's Foot or
  - Martin or
  - Information Engineering
- There are many other styles, and specific tools tailor styles.

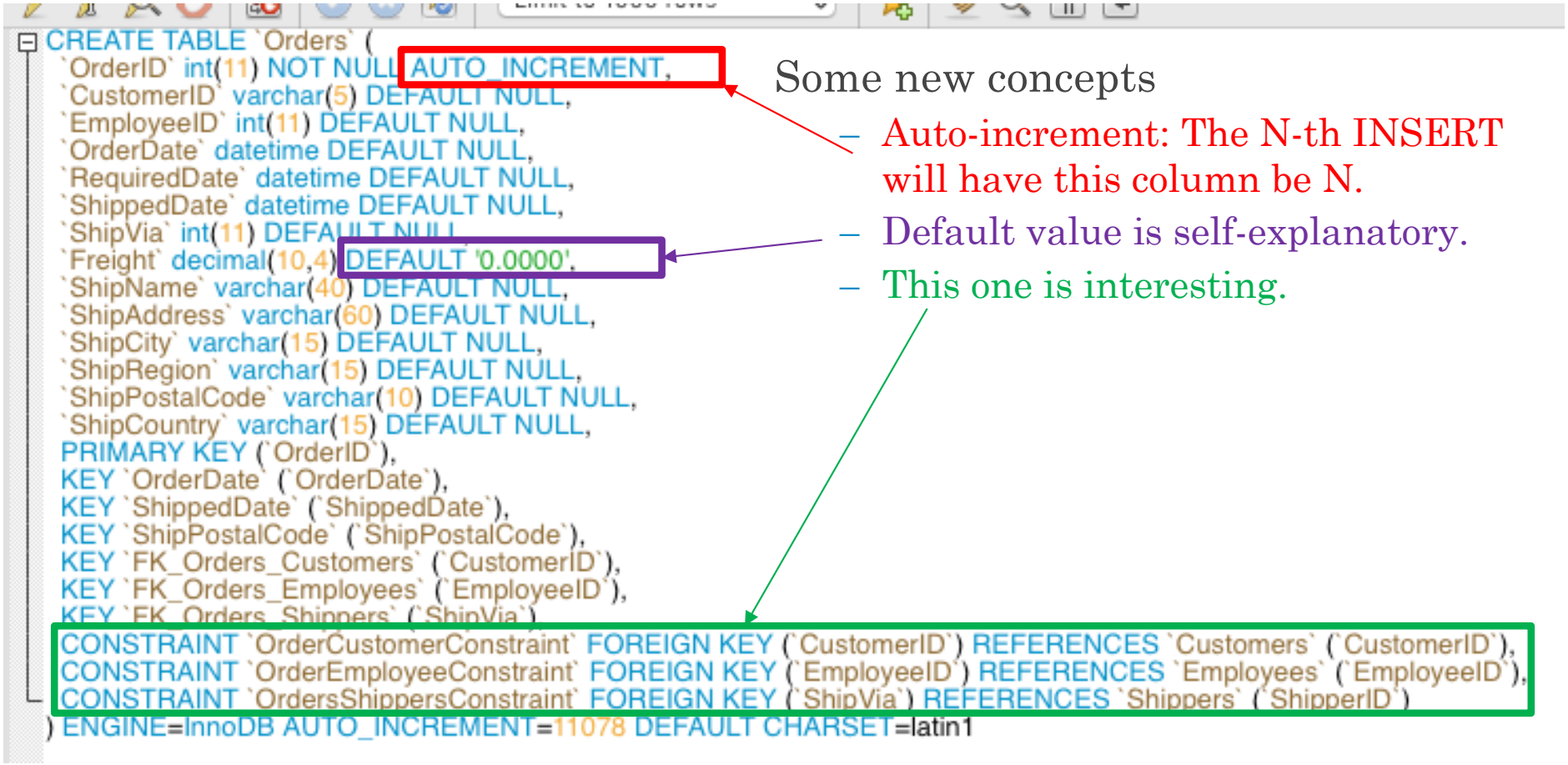
https://www.lucidchart.com/pages/er-diagrams/c?er=1

Some Additional Concepts (Chen Notation)



- Entity: Well, we know what this means.
- "Weak Entity:"
  - An entity whose existence depends on/is only defined relative to another entity.
  - OrderDetails is an example.
- "Associative Entity:"
  - An associative entity is an entity (e.g. has attributes) AND
  - Represents a relationship between sets of entities.
  - Usually used for many-to-many AND/OR the relationships has properties independent of the related entities.

Some DDL Examples



Keys in Data Models/Relational Model
(https://www.lucidchart.com/pages/er-diagrams/c?er=1)

Entity keys: Refers to an attribute that uniquely defines an entity in an entity set. Entity keys can be super, candidate or primary.

- **Super key:** A set of attributes (one or more) that together define an entity in an entity set.
- **Candidate key:** A minimal super key, meaning it has the least possible number of attributes to still be a super key. An entity set may have more than one candidate key.
- **Primary key:** A candidate key chosen by the database designer to uniquely identify the entity set.
- **Foreign key:** Identifies the relationship between entities.

Keys in Data Models/Relational Model

/c?er=1)

"Define" is a counter-intuitive way to phrase the concept. A more intuitive phrase is "uniquely identifies."

Define is an artifact of relational theory. If I know the key values, then I "know" what the other values will be.

efines an entity in an
ary.

together **define** an entity in an entity set.

- **Candidate key:** A minimal super key, meaning it has the least possible nu ntity set may have more tha

This definition is a little vague or circular.

- **Primary key:** A candidate ke y the database designer to uniquely identify the entity set.
- **Foreign key:** **Identifies the relationship between entities.**

Foreign Key (Molina et al).

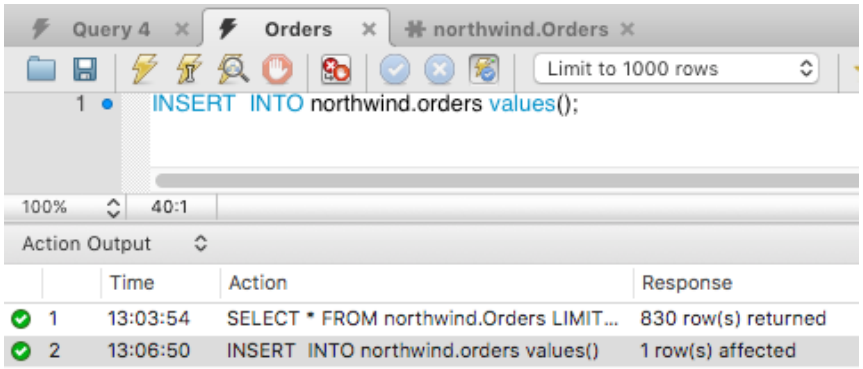
A foreign key constraint is an assertion that values for certain attributes must make sense. Recall, for instance, that in Example 2.21 we considered how to express in relational algebra the constraint that the producer "certificate number" for each movie was also the certificate number of some executive in the `MovieExec` relation.

In SQL we may declare an attribute or attributes of one relation to be a *foreign key*, referencing some attribute(s) of a second relation (possibly the same relation). The implication of this declaration is twofold:

1. The referenced attribute(s) of the second relation must be declared `UNIQUE` or the `PRIMARY KEY` for their relation. Otherwise, we cannot make the foreign-key declaration.

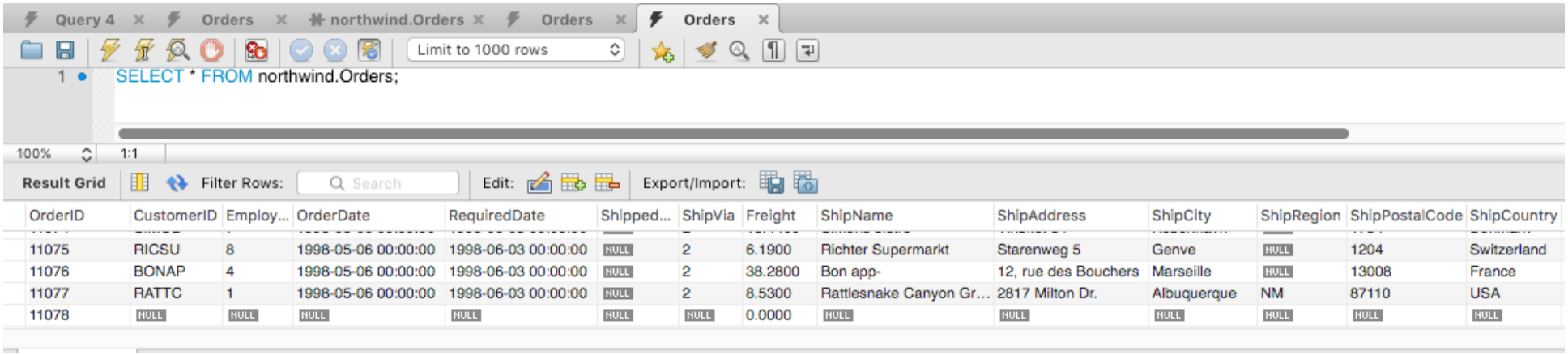
2. Values of the foreign key appearing in the first relation must also appear in the referenced attributes of some tuple. More precisely, let there be a foreign-key F that references set of attributes G of some relation. Suppose a tuple t of the first relation has non-`NULL` values in all the attributes of F; call the list of t's values in these attributes $t[F]$. Then in the referenced relation there must be some tuple s that agrees with $t[F]$ on the attributes G. That is, $s[G] = t[F]$.

That's Weird

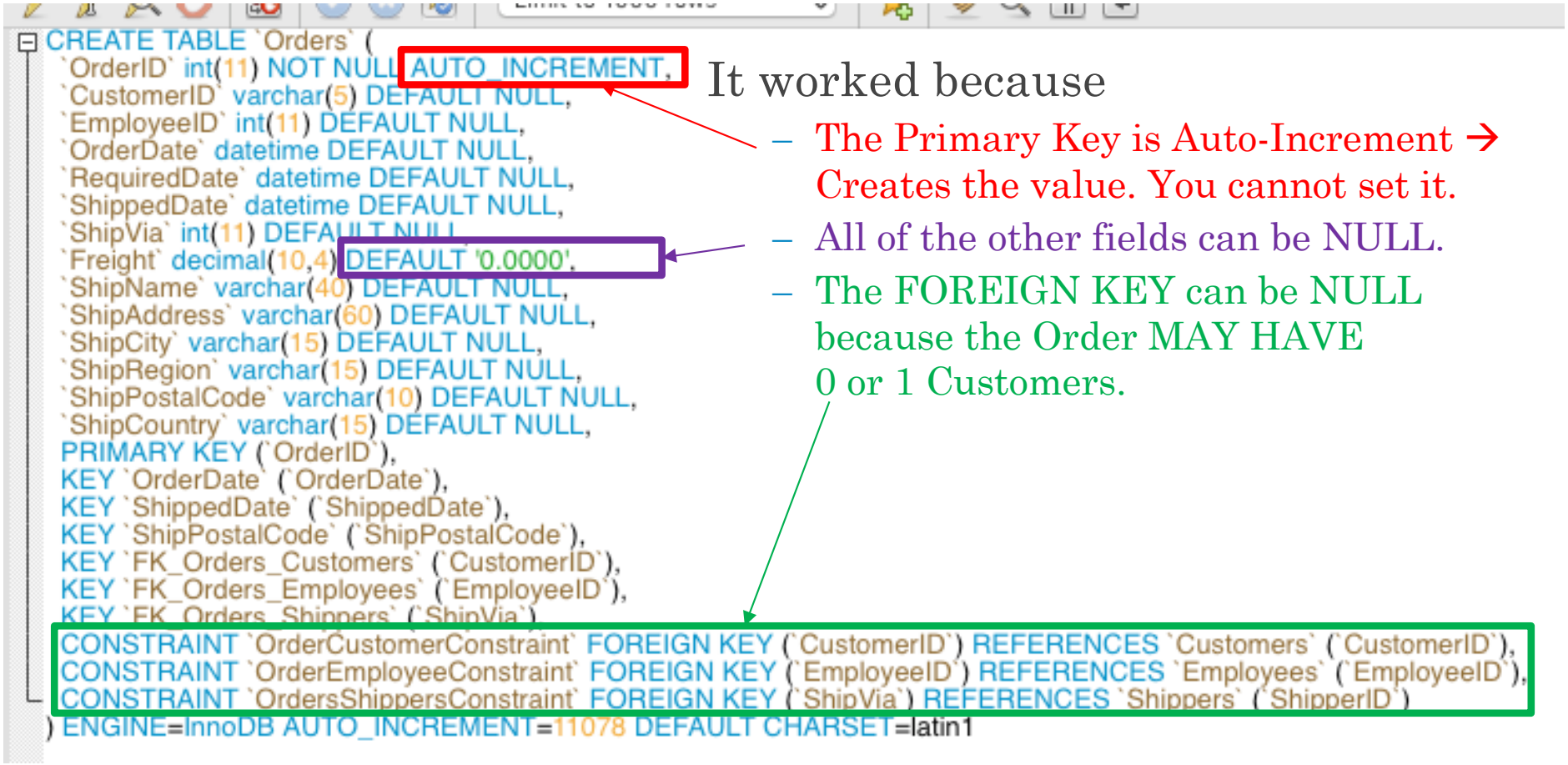


INSERT INTO orders VALUES() does not seem like it should work?

- What about the PRIMARY KEY?
- What about the FOREIGN KEY?



Not So Weird!



It worked because

- The Primary Key is Auto-Increment → Creates the value. You cannot set it.
- All of the other fields can be NULL.
- The FOREIGN KEY can be NULL because the Order MAY HAVE 0 or 1 Customers.

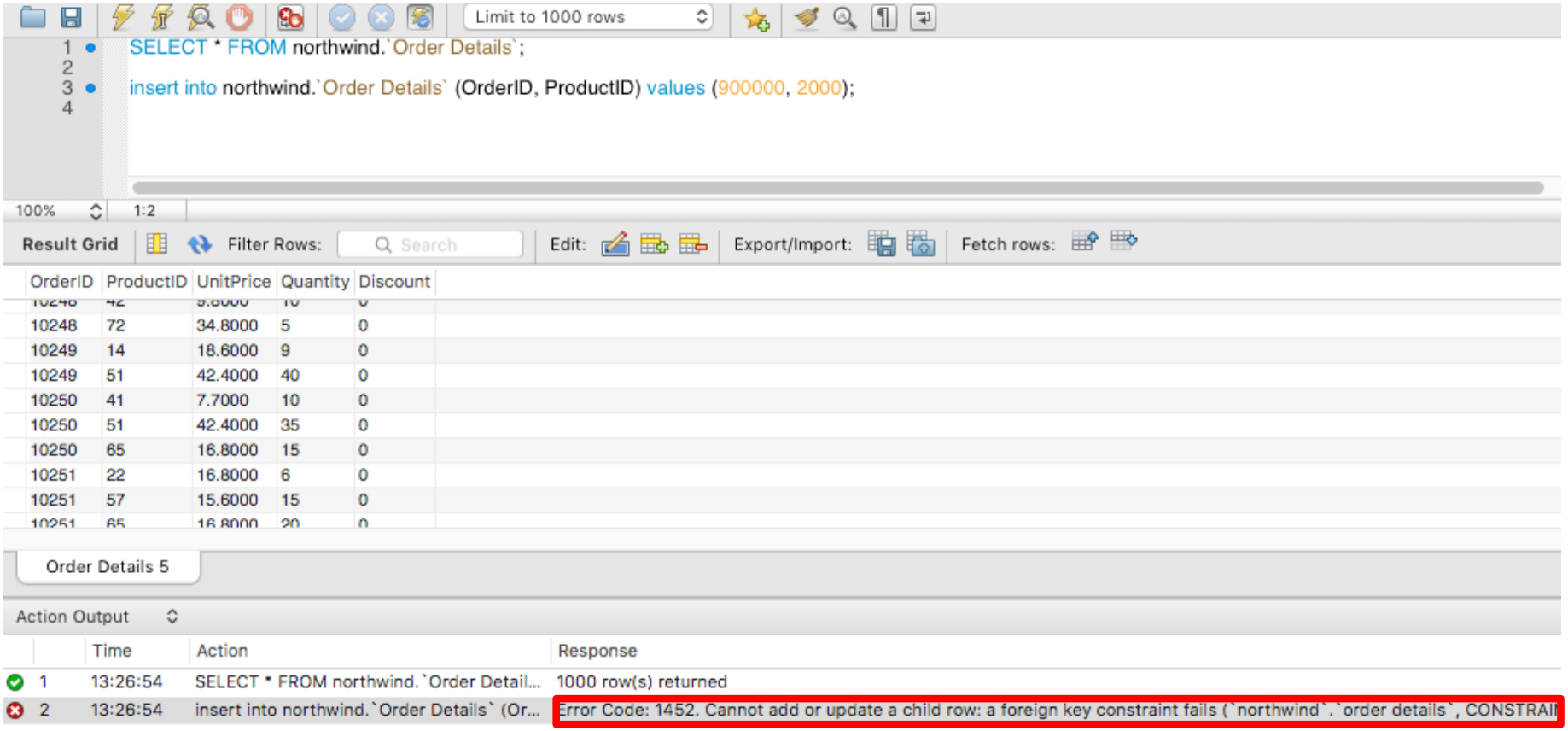
Order and OrderDetails

```
CREATE TABLE `Order Details` (
  `OrderID` int(11) NOT NULL,
  `ProductID` int(11) NOT NULL,
  `UnitPrice` decimal(10,4) NOT NULL DEFAULT '0.0000',
  `Quantity` smallint(2) NOT NULL DEFAULT '1',
  `Discount` double(8,0) NOT NULL DEFAULT '0',
  PRIMARY KEY (`OrderID`,`ProductID`),
  KEY `FK_Order_Details_Products` (`ProductID`),
  CONSTRAINT `FK_Order_Details_Orders` FOREIGN KEY (`OrderID`) REFERENCES `Orders` (`OrderID`),
  CONSTRAINT `FK_Order_Details_Products` FOREIGN KEY (`ProductID`) REFERENCES `Products` (`ProductID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

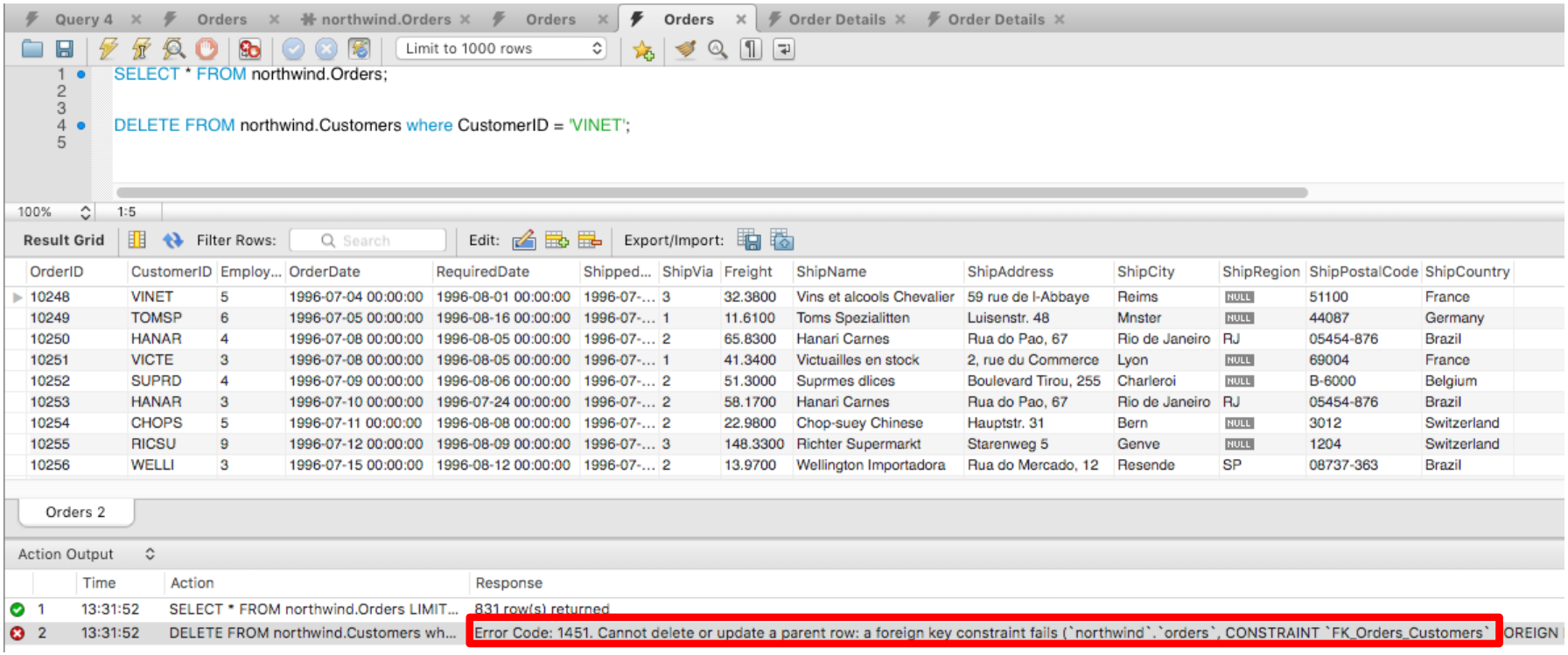
- For OrderDetails, both OrderID and ProductID
  - Are NOT NULL and
  - FOREIGN KEYS
- Which means that for INSERT and UPDATE, the value
  - Cannot be NULL
  - And MUST EXIST in the related table.

Let's Try

Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`northwind`.`order details`, CONSTRAINT `FK_Order_Details_Orders` FOREIGN KEY (`OrderID`) REFERENCES `Orders` (`OrderID`))



Let's Try Something Else.



Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails (`northwind`.`orders`, CONSTRAINT `FK_Orders_Customers` FOREIGN KEY (`CustomerID`) REFERENCES `Customers` (`CustomerID`))

Codd's 12 Rules

- **Rule 1 - Information Rule:** The data stored in a database, may it be user data or metadata, must be a value of some table cell. Everything in a database must be stored in a table format.
- **Rule 2 - Guaranteed Access Rule:** Every single data element (value) is guaranteed to be accessible logically with a combination of table-name, primary-key (row value), and attribute-name (column value). No other means, such as pointers, can be used to access data.
- **Rule 3 - Systematic Treatment of NULL Values:** The NULL values in a database must be given a systematic and uniform treatment. This is a very important rule because a NULL can be interpreted as one the following – data is missing, data is not known, or data is not applicable.
- **Rule 4 - Active Online Catalog:** The structure description of the entire database must be stored in an online catalog, known as **data dictionary**, which can be accessed by authorized users. Users can use the same query language to access the catalog which they use to access the database itself.
- **Rule 5 - Comprehensive Data Sub-Language Rule:** A database can only be accessed using a language having linear syntax that supports data definition, data manipulation, and transaction management operations. This language can be used directly or by means of some application. If the database allows access to data without any help of this language, then it is considered as a violation.
- **Rule 6 - View Updating Rule:** All the views of a database, which can theoretically be updated, must also be updatable by the system.

Codd's 12 Rules

- **Rule 7 - High-Level Insert, Update, and Delete Rule:** A database must support high-level insertion, updation, and deletion. This must not be limited to a single row, that is, it must also support union, intersection and minus operations to yield sets of data records.
- **Rule 8 - Physical Data Independence:** The data stored in a database must be independent of the applications that access the database. Any change in the physical structure of a database must not have any impact on how the data is being accessed by external applications.
- **Rule 9 - Logical Data Independence:** The logical data in a database must be independent of its user's view (application). Any change in logical data must not affect the applications using it. For example, if two tables are merged or one is split into two different tables, there should be no impact or change on the user application. This is one of the most difficult rule to apply.
- **Rule 10 - Integrity Independence:** A database must be independent of the application that uses it. All its integrity constraints can be independently modified without the need of any change in the application. This rule makes a database independent of the front-end application and its interface.
- **Rule 11 - Distribution Independence:** The end-user must not be able to see that the data is distributed over various locations. Users should always get the impression that the data is located at one site only. This rule has been regarded as the foundation of distributed database systems.
- **Rule 12 - Non-Subversion Rule:** If a system has an interface that provides access to low-level records, then the interface must not be able to subvert the system and bypass security and integrity constraints

Codd's 12 Rules

- **Rule 7 - High-Lev** updation, and del intersection and n
- **Rule 8 - Physica** applications that any impact on hov
- **Rule 9 - Logical Data Indepe** data in a database must be independent of its user's view (application). Any change must not affect the applications using it. For example, if two tables are merged or one is split into two different tables, there should be no impact or change on the user application. This is one of the most difficult rule to apply.
- **Rule 10 - Integrity Independence:** A database must be independent of the application that uses it. All its integrity constraints can be independently modified without the need of any change in the application. This rule makes a database independent of the front-end application and its interface.
- **Rule 11 - Distribution Independence:** The end-user must not be able to see that the data is distributed over various locations. Users should always get the impression that the data is located at one site only. This rule has been regarded as the foundation of distributed database systems.
- **Rule 12 - Non-Subversion Rule:** If a system has an interface that provides access to low-level records, then the interface must not be able to subvert the system and bypass security and integrity constraints

Lot's of different programmers, programming lots of programs at lot's of different times cannot MESS UP MY DATA.

More Sophisticated Constraints

```
CREATE TABLE IF NOT EXISTS Professor
(UNI varchar(8) ,
last_name  varchar(32),
first_name varchar(32) ,
iq int,
CONSTRAINT CHK_Person CHECK ((iq > 0 AND iq < 100) AND (NOT (last_name = 'Ferguson' AND iq > 50))),
PRIMARY KEY (UNI) );
```

- There are several types of CONSTRAINT commonly used
  - NOT NULL - Ensures that a column cannot have a NULL value
  - UNIQUE - Ensures that all values in a column are different
  - PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
  - FOREIGN KEY - Uniquely identifies a row/record in another table
  - CHECK - Ensures that all values in a column satisfies a specific condition
  - DEFAULT - Sets a default value for a column when no value is specified
  - INDEX - Use to create and retrieve data from the database very quickly
- But database product support is inconsistent for CHECK.

JOIN

Relationships

"How do I find a Customer's Orders?"

Query 7 | Customers | Orders

Limit to 1000 rows

```
1  SELECT
2     Customers.CustomerID as CustomerID, Customers.CompanyName as CompanyName, Customers.ContactName as ContactName,
3     Orders.OrderID as OrderID, Orders.OrderDate as OrderDate
4  FROM
5     Customers JOIN Orders
6  on
7     Customers.CustomerID = Orders.CustomerID
8  order by Orders.OrderDate ASC;
```

100% 31:8

Result Grid | Filter Rows: Search | Export:

CustomerID	CompanyName	ContactName	OrderID	OrderDate
VINET	Vins et alcools Chevalier	Paul Henriot	10248	1996-07-04 00:00:00
TOMSP	Toms Spezialitten	Karin Josephs	10249	1996-07-05 00:00:00
HANAR	Hanari Carnes	Mario Pontes	10250	1996-07-08 00:00:00
VICTE	Victuailles en stock	Mary Saveley	10251	1996-07-08 00:00:00
SUPRD	Suprmes dlices	Pascale Cartrain	10252	1996-07-09 00:00:00
HANAR	Hanari Carnes	Mario Pontes	10253	1996-07-10 00:00:00
CHOPS	Chop-suey Chinese	Yang Wang	10254	1996-07-11 00:00:00
RICSU	Richter Supermarkt	Michael Holz	10255	1996-07-12 00:00:00
WELLI	Wellington Importadora	Paula Parente	10256	1996-07-15 00:00:00
HILAA	HILARION-Abastos	Carlos Hernndez	10257	1996-07-16 00:00:00
ERNSH	Ernst Handel	Roland Mendel	10258	1996-07-17 00:00:00
CENTC	Centro comercial Moct…	Francisco Chang	10259	1996-07-18 00:00:00
QUEDE	Que Delcia	Bernardo Batista	10261	1996-07-19 00:00:00
OTTIK	Ottilies Kseladen	Henriette Pfalzheim	10260	1996-07-19 00:00:00

JOIN

- Remember TWO facts
  - Basic SQL statement structure.
    SELECT <columns> FROM <table> WHERE <expression>
  - SQL is an Algebra that allows combining statements.
- JOIN makes a table (temporary)
  - The attributes (columns) of the JOIN are the union of the tables' attributes.
  - If table A has N tuples and table B has M tuples, there are NxM possible tuples.
  - Include ONLY those tuples for which the ON clause is TRUE.
- This temporary table goes into the <table> slot in the standard SQL clause.
- There are several types of JOIN, which we will cover next lecture(s).

Homework 2

Homework 2

Define (data model) and implement (DDL) entities and relationships for

- Students
- Faculty
- Course
- Section (of course)
- Enrollment (enrolled, waitlisted)

- This should include
  - Modeling and implementing required/optional relationships, cardinality, etc.
    You can use whatever notation you want as long as clear.
  - Implementing constraints, except for CHECK.
  - Define a set of interesting user stories and show supporting SQL queries, data model chosen and constraints chosen.
  - Populate with sample/test data.
- Submission details
  - Date/Time: 29-Sep-2017, 23:59:59 EDT.
  - Data Model diagrams, with 3-4 pages of text explaining the model design choices, etc.
  - DDL
  - User stories you chose, supporting SQL statements and screen shot/recording of execution.

Homework 2

- This assignment is
  - Complex and a lot of work
  - Using Concepts you have just been introduced to.
- I do not expect you to get it all done or get it all correct.
- We will learn iteratively based on feedback and corrections.
- The IAs and I will help.
- I said in lecture 1 that HWs would be complex, hard and vague. **Figuring out what to do and what is important is part of the learning experience.**