Exam on 2018-11-13

Stewart Johnston
CIS 150 – Intro to Database Administration
NCMC
johnstons1@student.ncmich.edu

November 13, 2018

Contents

List of Figures

1 Short answers

1.1 Data independence

comes in two flavors: Physical data independence and Logical data independence. The principle function of data independence is to allow for changes without having to rewrite applications which use the data. Physical modifications would normally be made for performance reasons, but we don't want these changes to break queries. Logical changes would happen primarily when the conceptual scheme is altered. These might change some constraints, but shouldn't interfere with existing operations.

1.2 Primary keys

should be both stable and unique. It is important that any given row is always able to be referenced unambiguously, and that the method of reference never changes. To that end, primary keys should be either be made or chosen from immutable data. They also cannot be null.

1.3 DDL and DML

stand for Data Definition Language and Data Manipulation Language, respectively. They are two of the subsets of language defined for SQL. Data Definition Language has much to do with the creation and alteration of tables, especially of the physical data-types used for the information. Data Manipulation Langauge, meanwhile, is related to retrieving and editing tuples and records. If the initial database design is solid and meets the needs of the users, DDL may go untouched for years. DML, on the other hand, will certainly see frequent use in any living database.

```
/* Data Definiton Language example */
CREATE TABLE Parts(
id      INTEGER     PRIMARY KEY,
name    VARCHAR(50) NOT NULL,
length  FLOAT       NOT NULL,
width   FLOAT       NOT NULL,
);
```

```sql
/* Data Manipulation Language example */
SELECT name, length, width
FROM Parts
WHERE length > 12;
```

1.4 Foreign Keys

are the primary vehicle for describing and constraining relationships in our data. They allow tuples to reference other tuples, either in the same or in a different table. They provide database administrators a means to keep data consistent, both in the form of normalization, and in the form of referential integrity. Referential integrity is a constraint for the RDBMS which tells it to forbid the deletion of records which are being referenced by other records, meaning that to delete a record which is referenced, one must prune or re-assign all records from all tables which reference it in one shot.

1.5 "Three levels of schema" architecture

involves an external schema, conceptual schema, and internal schema. Several external schema can exist for any given database, which describe how users see the data. Conceptual schema describes the relationships between entities in a database. The internal or physical schema describes the datatypes and other physical storage requirements for the data.

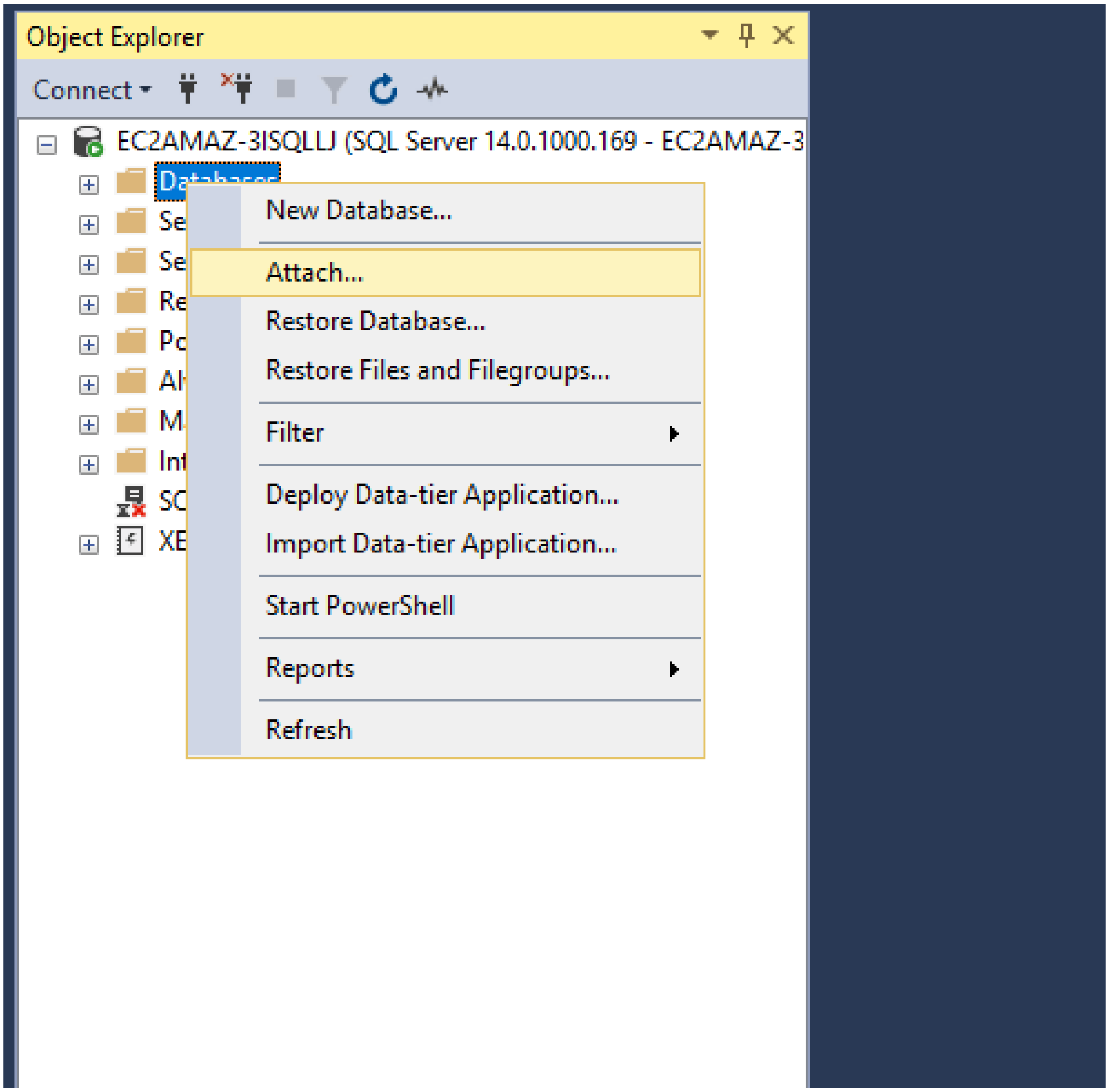
2 Hands on with the server

2.1 Attaching using SSMS

2.1.1

Right-click on Databases, select Attach... See figure 1

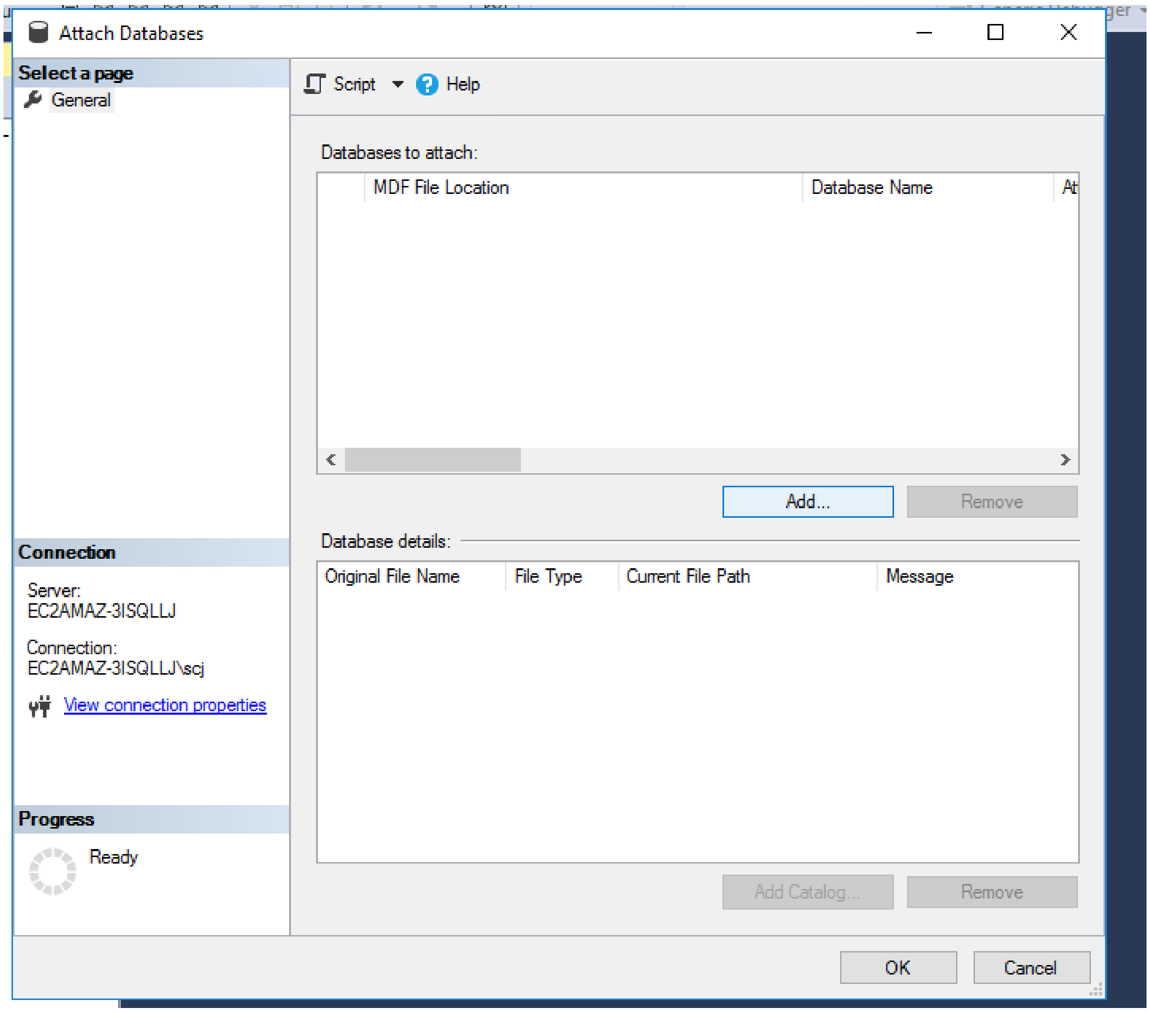
Figure 1: Per 2.1.1



2.1.2

In Databases to Attach, choose Add See figure 2

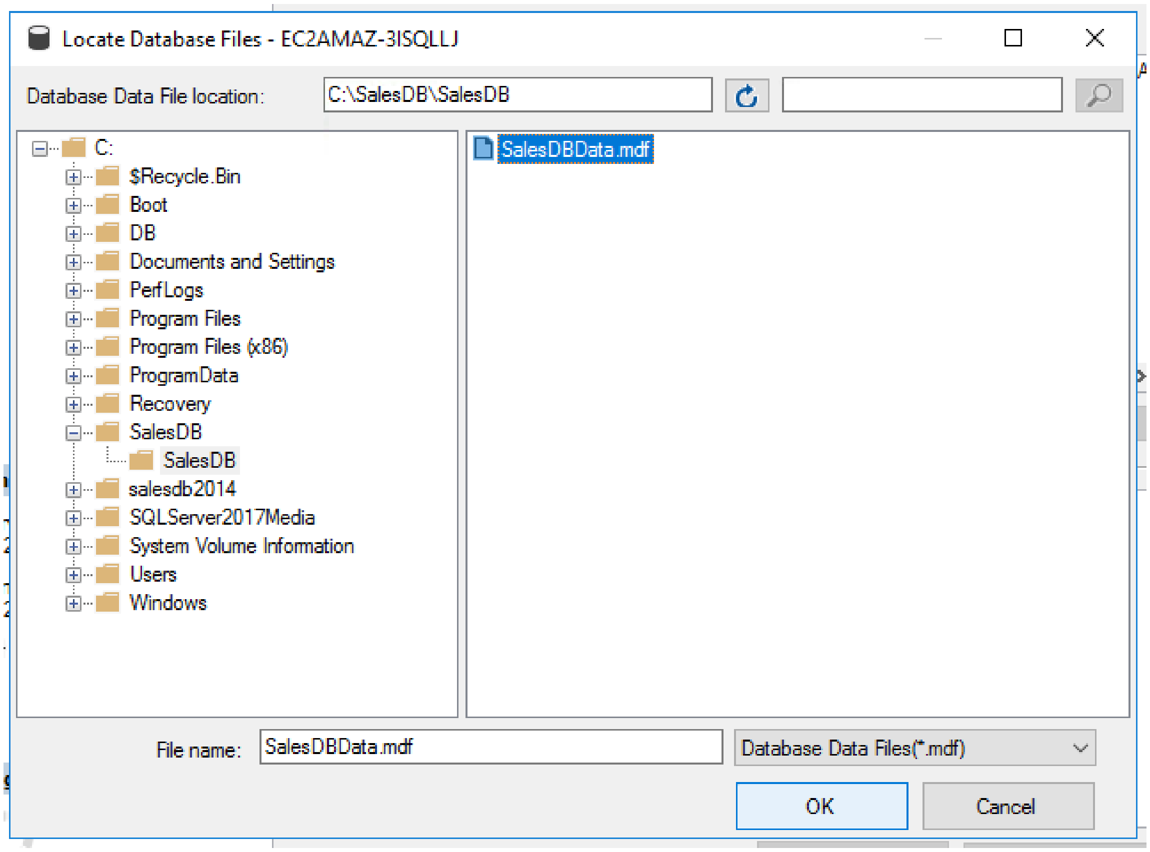
Figure 2: Per 2.1.2



2.1.3

This opens a file explorer window for browsing, find the database file. See figure 3

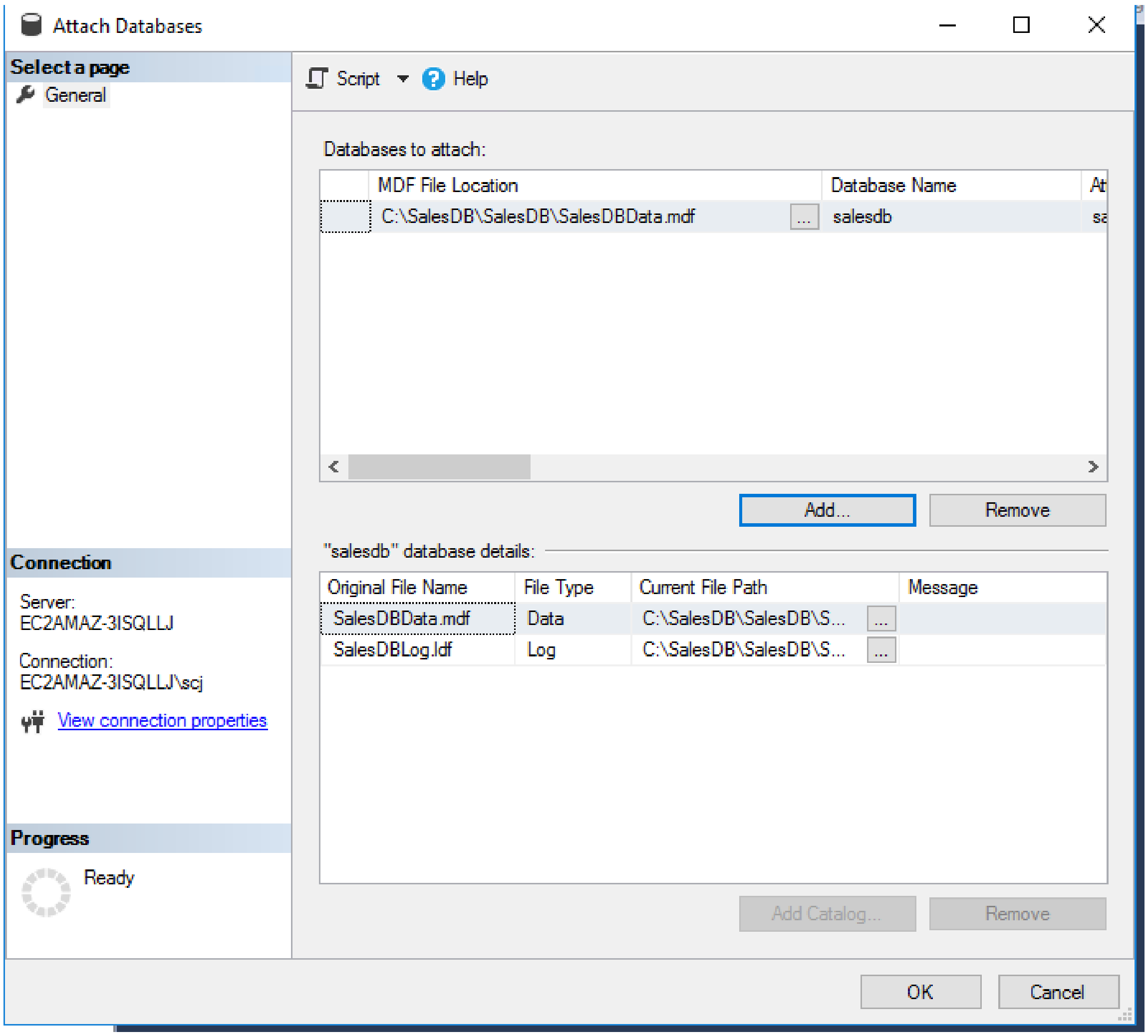
Figure 3: Per 2.1.3



2.1.4

If there is a log file with the database, it will grab that as well. Click OK. See figure 4

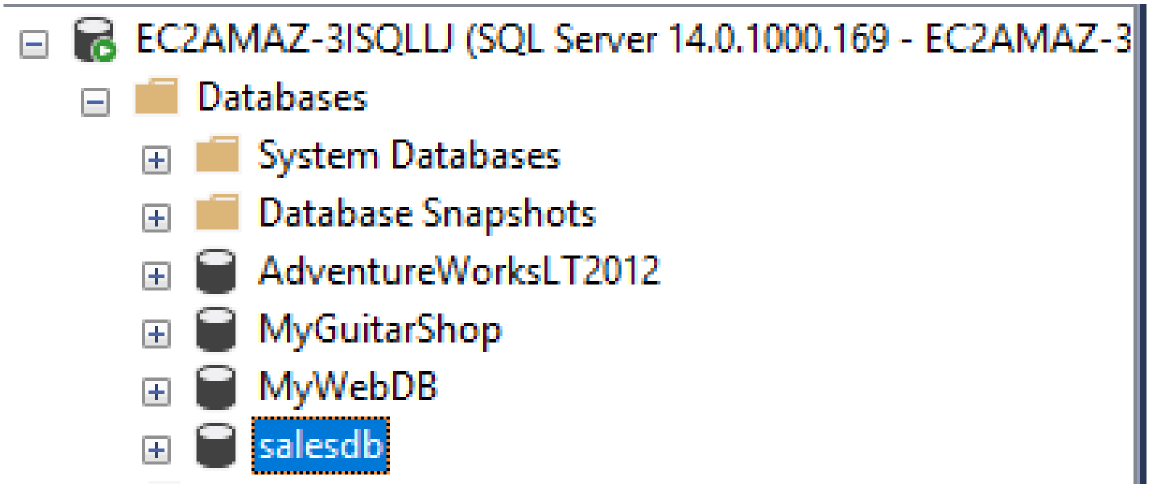
Figure 4: Per 2.1.4



2.1.5

Thusly a database is attached under the master. See figure 5

Figure 5: Per 2.1.5

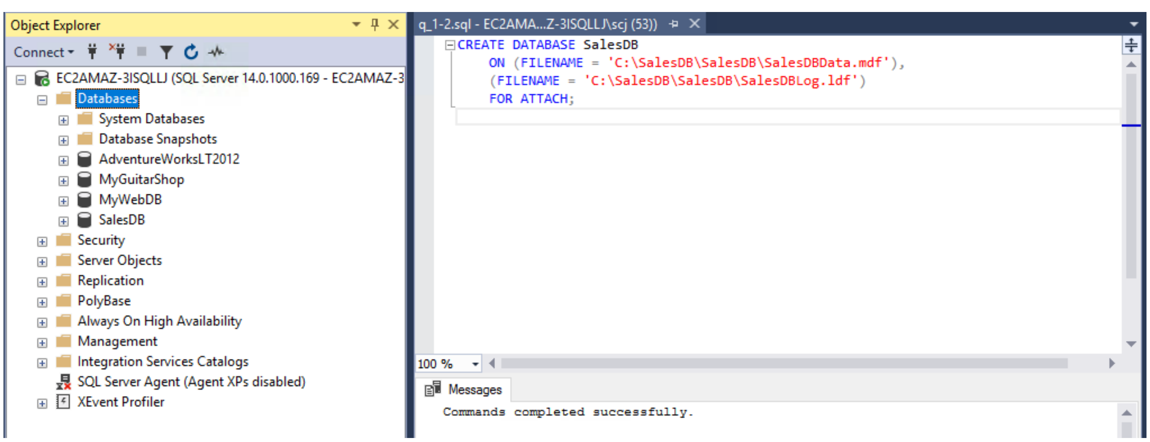


2.2 Attaching a database using DDL

See figure 6

```
CREATE DATABASE SalesDB
    ON (FILENAME = 'C:\SalesDB\SalesDB\SalesDBData.mdf'),
    (FILENAME = 'C:\SalesDB\SalesDB\SalesDBLog.ldf')
    FOR ATTACH;
```

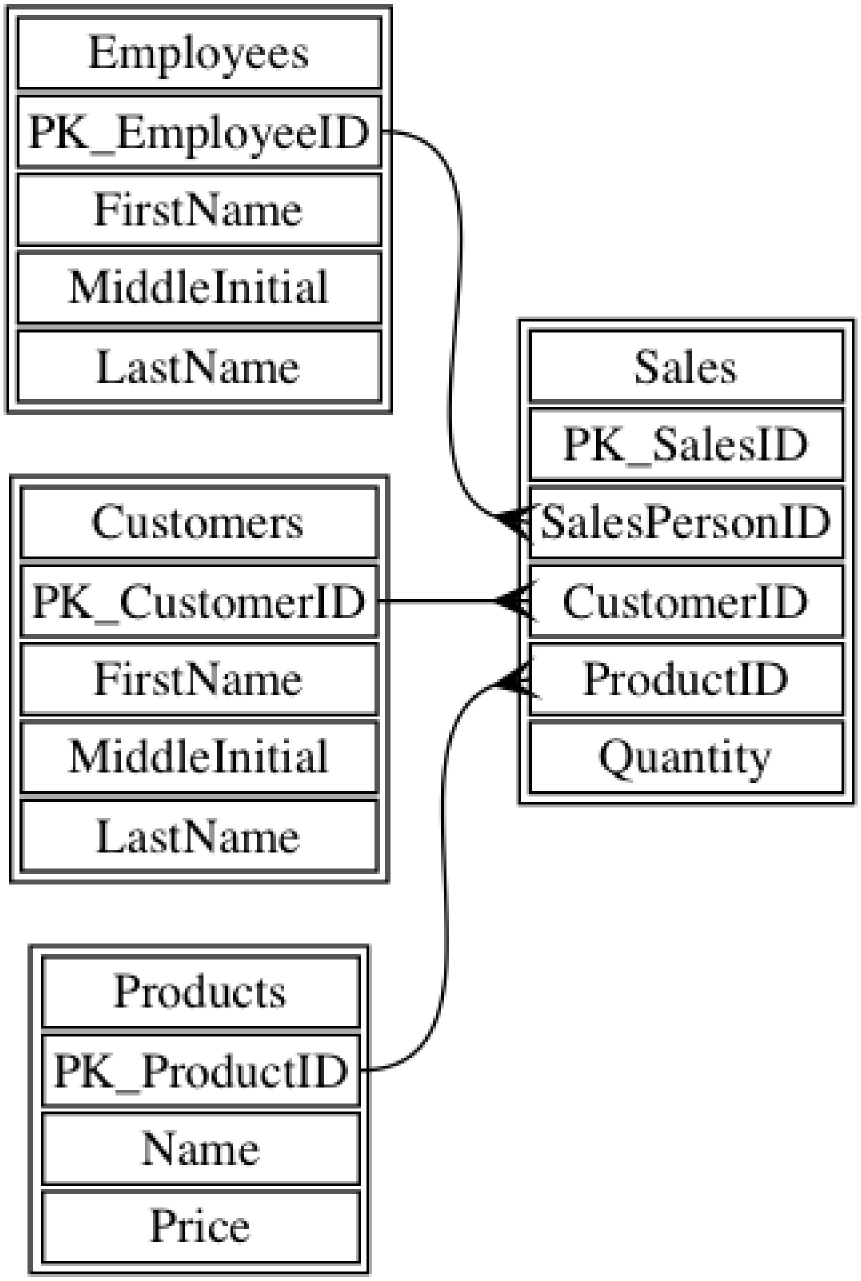
Figure 6: Per 2.2



2.3 Entity relationship diagram

showing Primary Keys as 'PK_', foreign keys with cardinality as crow's foot, and other attributes as part of the tables. See figure 7

Figure 7: Per 2.3



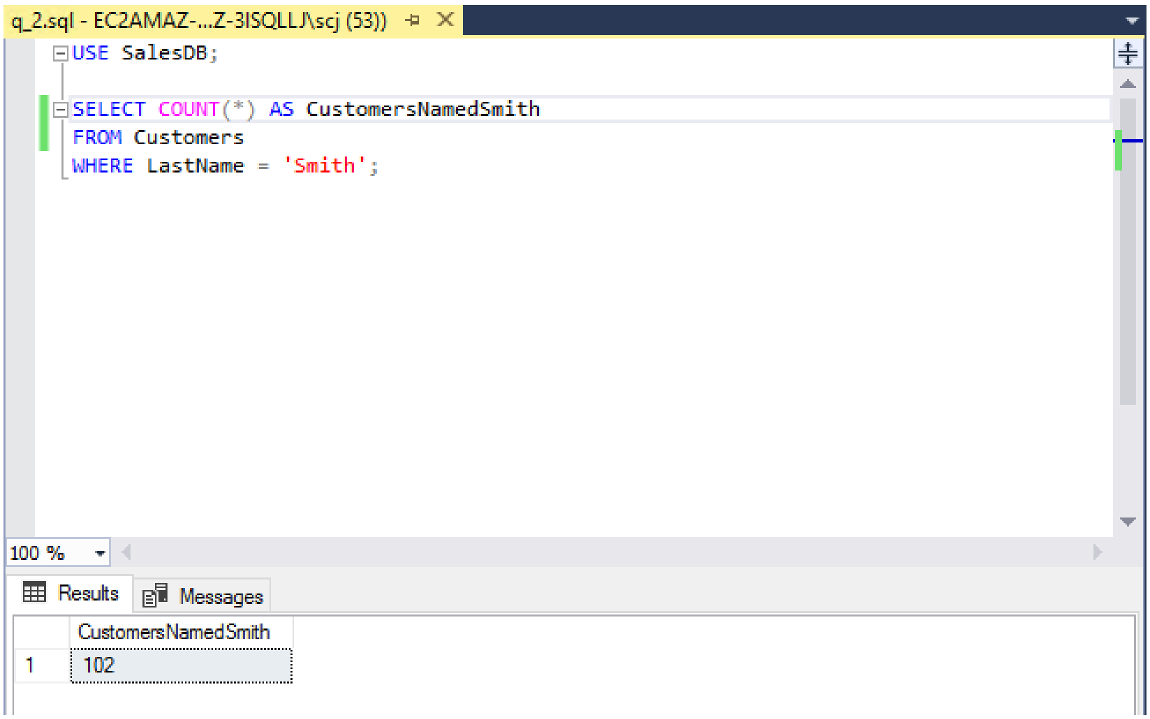
2.4 Count of customers named "Smith"

See figure 8

```
USE SalesDB;

SELECT COUNT(*) AS CustomersNamedSmith
FROM Customers
WHERE LastName = 'Smith';
```

Figure 8: Per 2.4



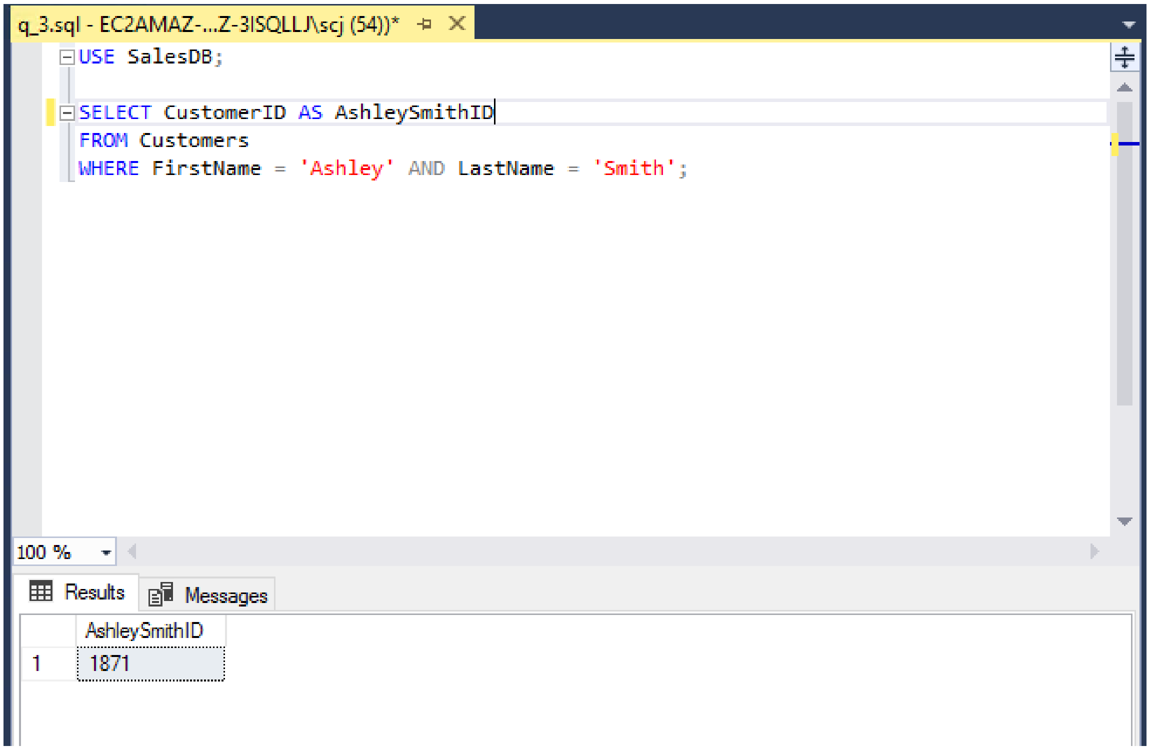
2.5 Ashley Smith's CustomerID

See figure 9

```
USE SalesDB;

SELECT CustomerID AS AshleySmithID
FROM Customers
WHERE FirstName = 'Ashley' AND LastName = 'Smith';
```

Figure 9: Per 2.5



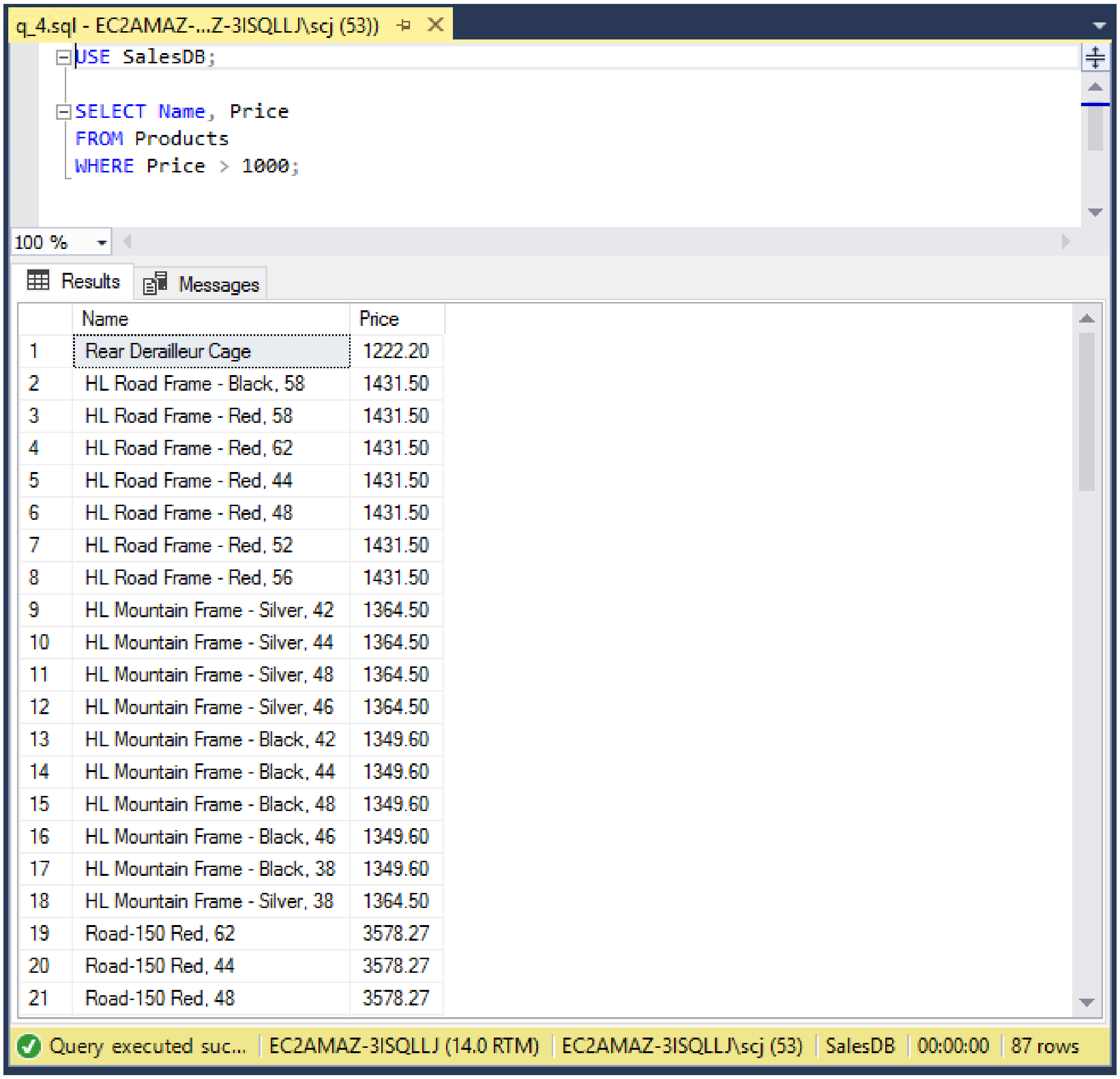
2.6 Names and prices of products more than $1000

See figure 10

```
USE SalesDB;

SELECT Name, Price
FROM Products
WHERE Price > 1000;
```

Figure 10: Per 2.6



q_4.sql - EC2AMAZ-...Z-3ISQLLJ\scj (53))

```sql
USE SalesDB;

SELECT Name, Price
FROM Products
WHERE Price > 1000;
```

100 %

Results | Messages

	Name	Price
1	Rear Derailleur Cage	1222.20
2	HL Road Frame - Black, 58	1431.50
3	HL Road Frame - Red, 58	1431.50
4	HL Road Frame - Red, 62	1431.50
5	HL Road Frame - Red, 44	1431.50
6	HL Road Frame - Red, 48	1431.50
7	HL Road Frame - Red, 52	1431.50
8	HL Road Frame - Red, 56	1431.50
9	HL Mountain Frame - Silver, 42	1364.50
10	HL Mountain Frame - Silver, 44	1364.50
11	HL Mountain Frame - Silver, 48	1364.50
12	HL Mountain Frame - Silver, 46	1364.50
13	HL Mountain Frame - Black, 42	1349.60
14	HL Mountain Frame - Black, 44	1349.60
15	HL Mountain Frame - Black, 48	1349.60
16	HL Mountain Frame - Black, 46	1349.60
17	HL Mountain Frame - Black, 38	1349.60
18	HL Mountain Frame - Silver, 38	1364.50
19	Road-150 Red, 62	3578.27
20	Road-150 Red, 44	3578.27
21	Road-150 Red, 48	3578.27

Query executed suc... | EC2AMAZ-3ISQLLJ (14.0 RTM) | EC2AMAZ-3ISQLLJ\scj (53) | SalesDB | 00:00:00 | 87 rows

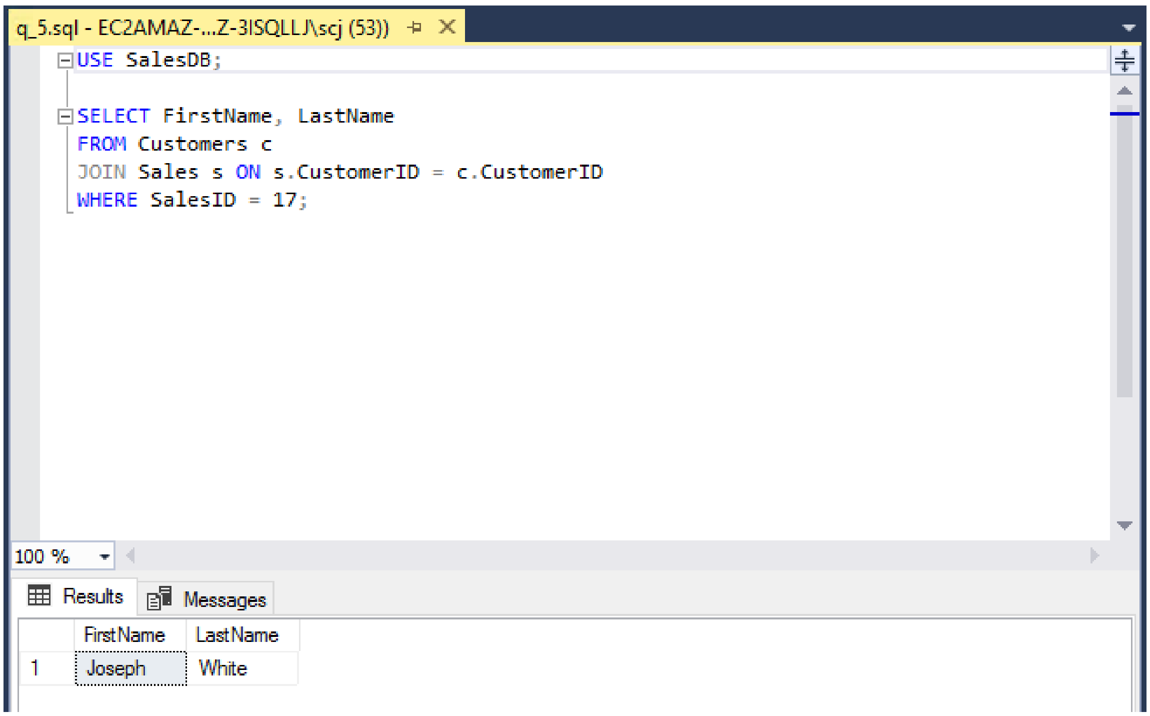
2.7 Name of customer from sale 17

See figure 11

```
USE SalesDB;

SELECT FirstName, LastName
FROM Customers c
JOIN Sales s ON s.CustomerID = c.CustomerID
WHERE SalesID = 17;
```

Figure 11: Per 2.7



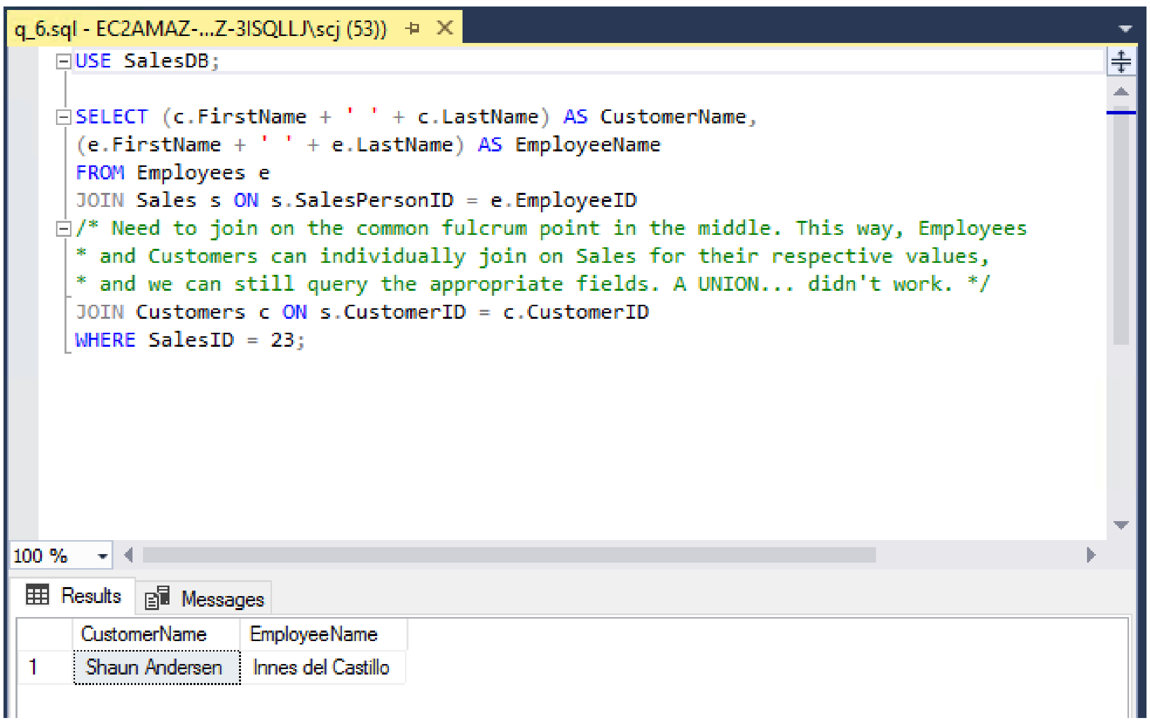
2.8 Customer and Employee from sale 23

See figure 12

```
USE SalesDB;

SELECT (c.FirstName + '␣' + c.LastName) AS CustomerName,
(e.FirstName + '␣' + e.LastName) AS EmployeeName
FROM Employees e
JOIN Sales s ON s.SalesPersonID = e.EmployeeID
/* Need to join on the common fulcrum point in the middle. This way, Employees
* and Customers can individually join on Sales for their respective values,
* and we can still query the appropriate fields. A UNION... didn't work. */
JOIN Customers c ON s.CustomerID = c.CustomerID
WHERE SalesID = 23;
```

Figure 12: Per 2.8



2.9 Customer, Employee, and product details of Sale 31

See figure 13

```
USE SalesDB;

SELECT (c.FirstName + '␣' + c.Lastname) AS CustomerName,
(e.FirstName + '␣' + e.LastName) AS EmployeeName,
ProductName,
ProductCost,
TotalPurchase
FROM ( --Subquery for the products-sales join
        SELECT
        p.Name AS ProductName,
        p.Price AS ProductCost,
        (p.Price * s.Quantity) AS TotalPurchase
        FROM Products p
        JOIN Sales s ON s.ProductID = p.ProductID
        WHERE SalesID = 31
) AS PurchasedProductDetails,
Customers c
JOIN Sales s ON c.CustomerID = s.CustomerID
JOIN Employees e ON e.EmployeeID = s.SalesPersonID
WHERE SalesID = 31;
```

Figure 13: Per 2.9