University of Salford MANCHESTER

Week 3: Introduction to Databases & SQL

School of Science, Engineering and Environment
University of Salford

Dr. Surbhi B. Khan

Advanced Databases

Learning Outcomes

- Understand the reasons for using a database and how using related tables helps you avoid the problems of using lists
- Learn the purpose of the database management system (DBMS)
- Provide an introduction to SQL
- Learn how to use SELECT queries to select data from a database
- Understand how to use predicates, operators and wildcards in queries
- Understand how to use DISTINCT, ORDER BY and TOP in SELECT queries
- Understand how to work with NULL values

Why to Study Databases?

- Databases are an ***essential*** component of life in modern society.
- We interact with databases in many daily activates, such as:-
  - Banking transactions: open an account, deposit, withdraw, transfer, etc.
  - Hotel or airline reservation.
  - E-commerce, buy or sell online.
  - Search for a book in the library.

Types of Databases and Database Applications

- **Traditional database applications**
  - Store textual or numeric information

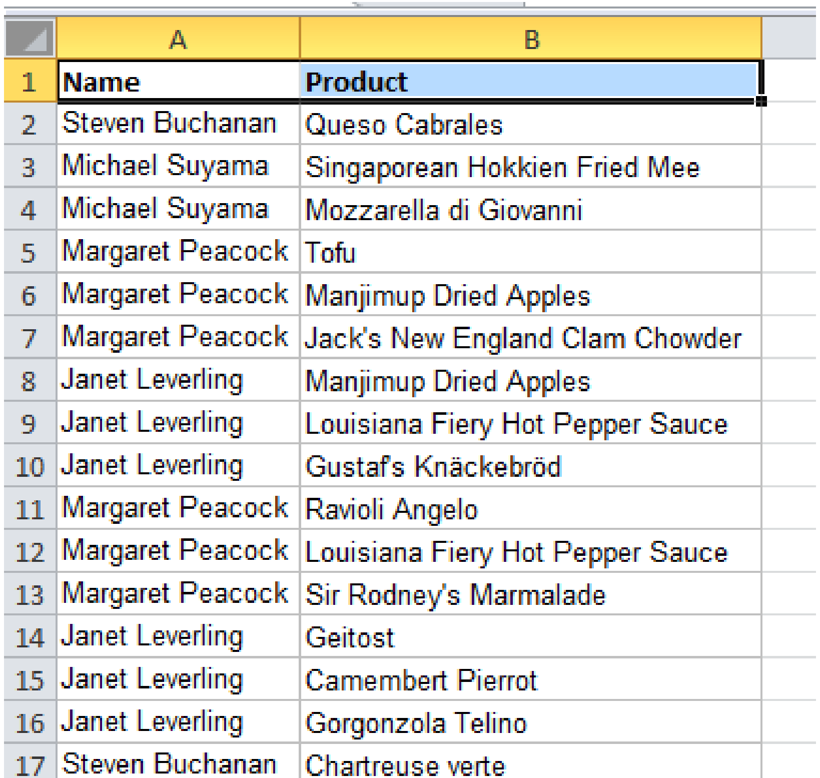
    Sales system, Cash handling system, reservation system

    Amazon.com- Example of large database application
- **Multimedia databases**
  - Store images, audio clips, and video streams digitally
- **Geographic information systems (GIS)**
  - Store and analyze maps, weather data, and satellite images

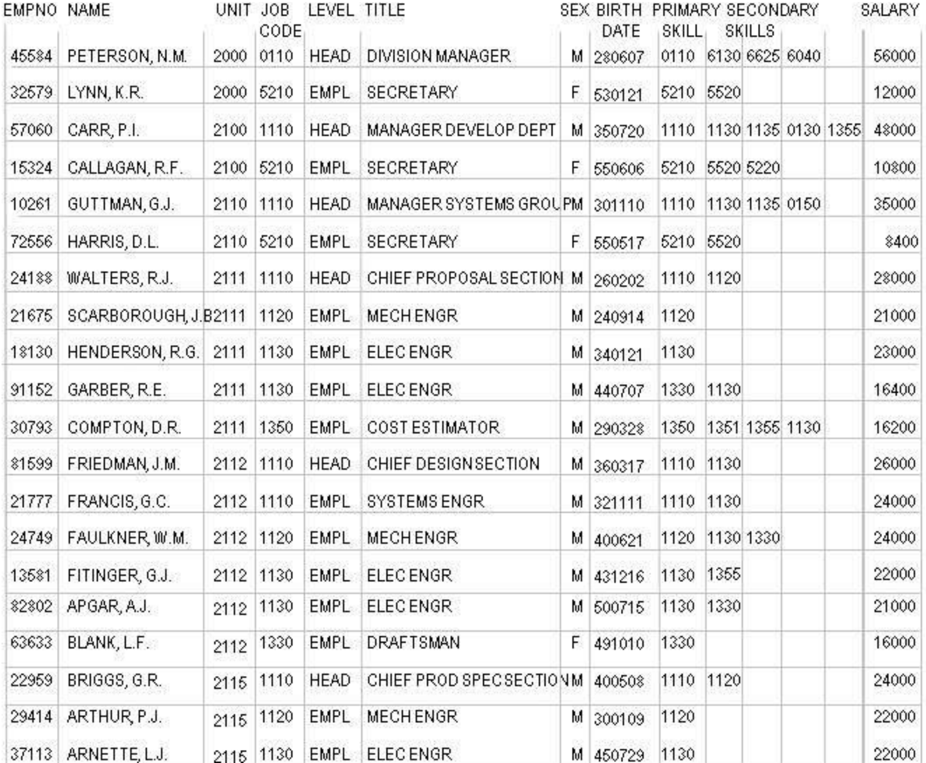
Types of Databases and Database Applications (cont'd.)

- **Data warehouses and online analytical processing (OLAP) systems**
  - Extract and analyze useful business information from very large databases
  - Support decision making
- **Real-time and active database technology**
  - Control industrial and manufacturing processes

Lists / Flat Files

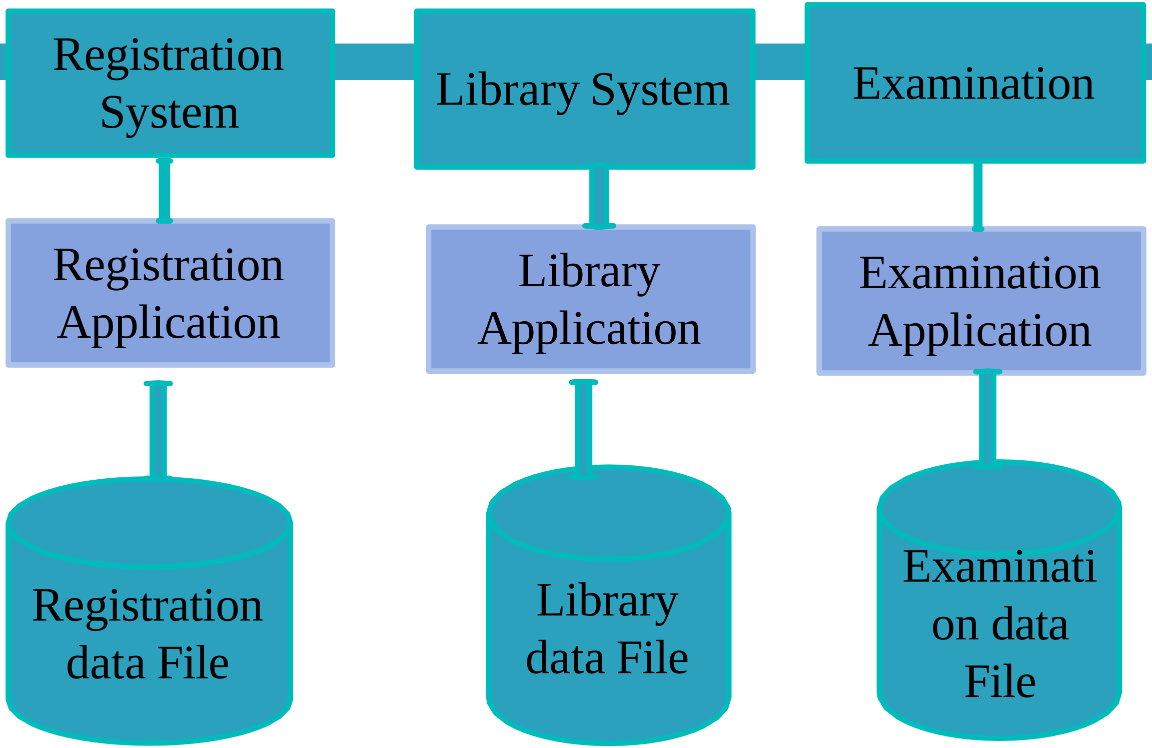


	A	B
1	**Name**	**Product**
2	Steven Buchanan	Queso Cabrales
3	Michael Suyama	Singaporean Hokkien Fried Mee
4	Michael Suyama	Mozzarella di Giovanni
5	Margaret Peacock	Tofu
6	Margaret Peacock	Manjimup Dried Apples
7	Margaret Peacock	Jack's New England Clam Chowder
8	Janet Leverling	Manjimup Dried Apples
9	Janet Leverling	Louisiana Fiery Hot Pepper Sauce
10	Janet Leverling	Gustaf's Knäckebröd
11	Margaret Peacock	Ravioli Angelo
12	Margaret Peacock	Louisiana Fiery Hot Pepper Sauce
13	Margaret Peacock	Sir Rodney's Marmalade
14	Janet Leverling	Geitost
15	Janet Leverling	Camembert Pierrot
16	Janet Leverling	Gorgonzola Telino
17	Steven Buchanan	Chartreuse verte



EMPNO	NAME	UNIT	JOB CODE	LEVEL	TITLE	SEX	BIRTH DATE	PRIMARY SKILL	SECONDARY SKILLS				SALARY
45584	PETERSON, N.M.	2000	0110	HEAD	DIVISION MANAGER	M	280607	0110	6130	6625	6040		56000
32579	LYNN, K.R.	2000	5210	EMPL	SECRETARY	F	530121	5210	5520				12000
57060	CARR, P.I.	2100	1110	HEAD	MANAGER DEVELOP DEPT	M	350720	1110	1130	1135	0130	1355	48000
15324	CALLAGAN, R.F.	2100	5210	EMPL	SECRETARY	F	550606	5210	5520	5220			10800
10261	GUTTMAN, G.J.	2110	1110	HEAD	MANAGER SYSTEMS GROUP	M	301110	1110	1130	1135	0150		35000
72556	HARRIS, D.L.	2110	5210	EMPL	SECRETARY	F	550517	5210	5520				8400
24188	WALTERS, R.J.	2111	1110	HEAD	CHIEF PROPOSAL SECTION	M	260202	1110	1120				28000
21675	SCARBOROUGH, J.B	2111	1120	EMPL	MECH ENGR	M	240914	1120					21000
18130	HENDERSON, R.G.	2111	1130	EMPL	ELEC ENGR	M	340121	1130					23000
91152	GARBER, R.E.	2111	1130	EMPL	ELEC ENGR	M	440707	1330	1130				16400
30793	COMPTON, D.R.	2111	1350	EMPL	COST ESTIMATOR	M	290328	1350	1351	1355	1130		16200
81599	FRIEDMAN, J.M.	2112	1110	HEAD	CHIEF DESIGN SECTION	M	360317	1110	1130				26000
21777	FRANCIS, G.C.	2112	1110	EMPL	SYSTEMS ENGR	M	321111	1110	1130				24000
24749	FAULKNER, W.M.	2112	1120	EMPL	MECH ENGR	M	400621	1120	1130	1330			24000
13581	FITINGER, G.J.	2112	1130	EMPL	ELEC ENGR	M	431216	1130	1355				22000
82802	APGAR, A.J.	2112	1130	EMPL	ELEC ENGR	M	500715	1130	1330				21000
63633	BLANK, L.F.	2112	1330	EMPL	DRAFTSMAN	F	491010	1330					16000
22959	BRIGGS, G.R.	2115	1110	HEAD	CHIEF PROD SPEC SECTION	M	400508	1110	1120				24000
29414	ARTHUR, P.J.	2115	1120	EMPL	MECH ENGR	M	300109	1120					22000
37113	ARNETTE, L.J.	2115	1130	EMPL	ELEC ENGR	M	450729	1130					22000

File Processing (FPS)



- Program data dependence

File Processing (FPS)

Registration
Reg. No
Name
Father Name
Phone
Address
Class

Library
Reg. No
Name
Father Name
Book issued
Fine

Examination
Reg. No
Name
Address
Class
Semester
Grade

Note redundant information and discuss problems

Problems with Lists: Redundancy

In a list, each row is intended to stand on its own. As a result, the same information may be entered several times.

- For Example: A list of Projects may include the Project Manager’s Name, ID, and Phone Extension. If a particular person is managing 10 projects, his/her information would have to be entered 10 times.

Problems with Lists: Multiple Themes

In a list, each row may contain information on more than one theme. As a result, needed information may appear in the lists only if information on other themes is also present.

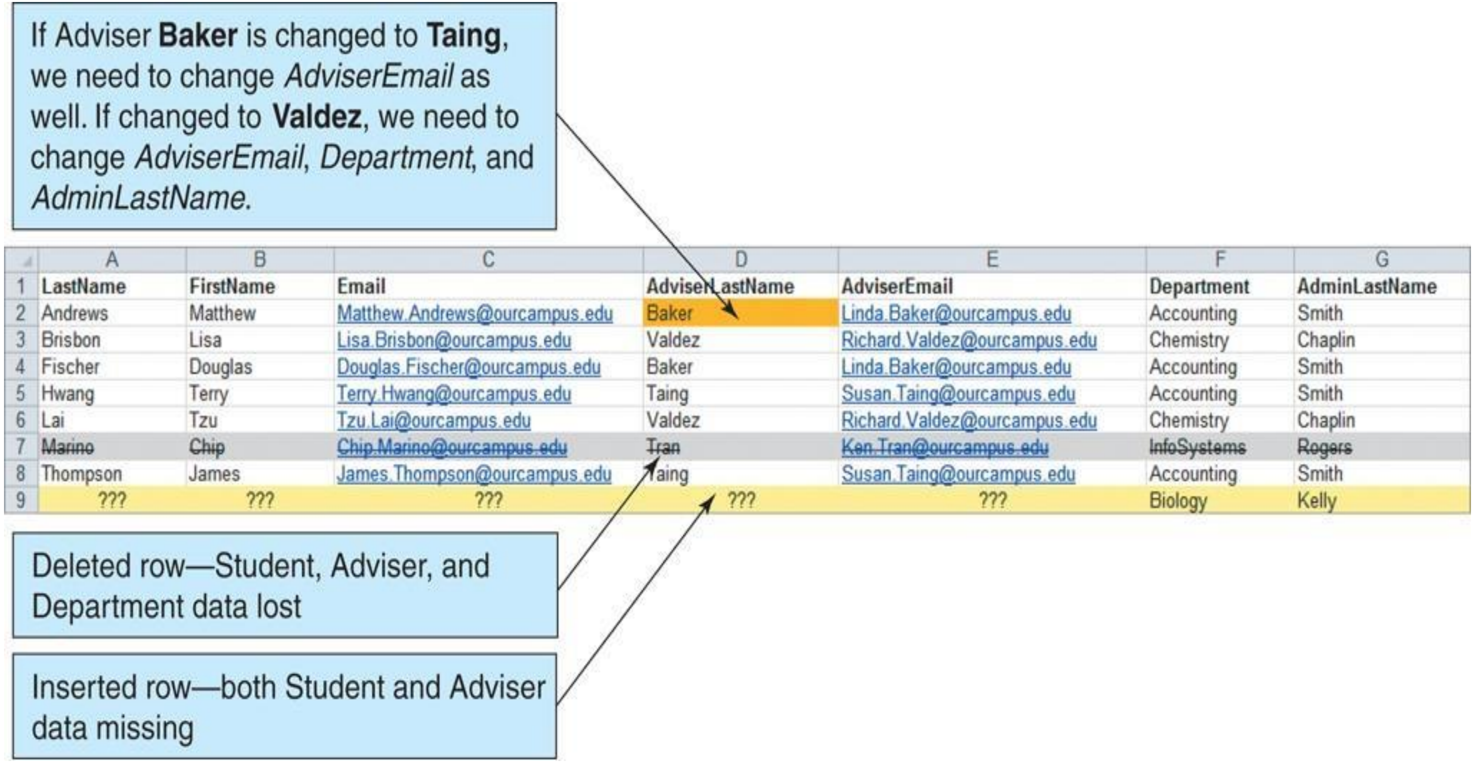
- For Example: A list of Projects may include Project Manager information (Name, ID, and Phone Extension) and Project information (Name, ID, StartDate, Budget) in the same row.

List Modification Issues

Redundancy and multiple themes create modification problems:

- Deletion problems
- Update problems
- Insertion problems

List Modification Issues



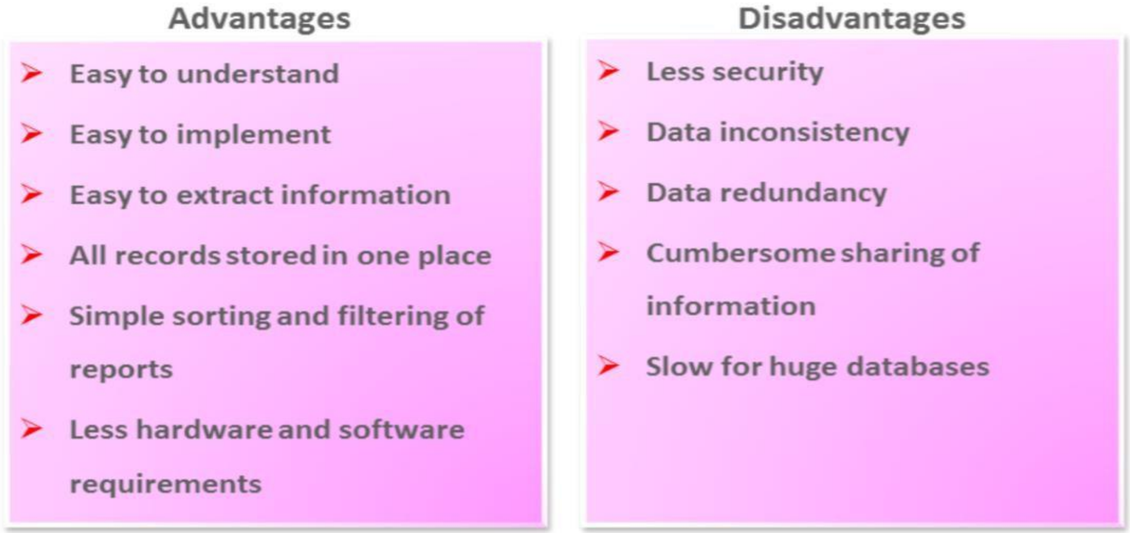
If Adviser **Baker** is changed to **Taing**, we need to change *AdviserEmail* as well. If changed to **Valdez**, we need to change *AdviserEmail*, *Department*, and *AdminLastName*.

	A	B	C	D	E	F	G
1	**LastName**	**FirstName**	**Email**	**AdviserLastName**	**AdviserEmail**	**Department**	**AdminLastName**
2	Andrews	Matthew	Matthew.Andrews@ourcampus.edu	Baker	Linda.Baker@ourcampus.edu	Accounting	Smith
3	Brisbon	Lisa	Lisa.Brisbon@ourcampus.edu	Valdez	Richard.Valdez@ourcampus.edu	Chemistry	Chaplin
4	Fischer	Douglas	Douglas.Fischer@ourcampus.edu	Baker	Linda.Baker@ourcampus.edu	Accounting	Smith
5	Hwang	Terry	Terry.Hwang@ourcampus.edu	Taing	Susan.Taing@ourcampus.edu	Accounting	Smith
6	Lai	Tzu	Tzu.Lai@ourcampus.edu	Valdez	Richard.Valdez@ourcampus.edu	Chemistry	Chaplin
7	~~Marino~~	~~Chip~~	~~Chip.Marino@ourcampus.edu~~	~~Tran~~	~~Ken.Tran@ourcampus.edu~~	~~InfoSystems~~	~~Rogers~~
8	Thompson	James	James.Thompson@ourcampus.edu	Taing	Susan.Taing@ourcampus.edu	Accounting	Smith
9	???	???	???	???	???	Biology	Kelly

Deleted row—Student, Adviser, and Department data lost

Inserted row—both Student and Adviser data missing

Advantages and disadvantages of flat file database



Advantages

- Easy to understand
- Easy to implement
- Easy to extract information
- All records stored in one place
- Simple sorting and filtering of reports
- Less hardware and software requirements

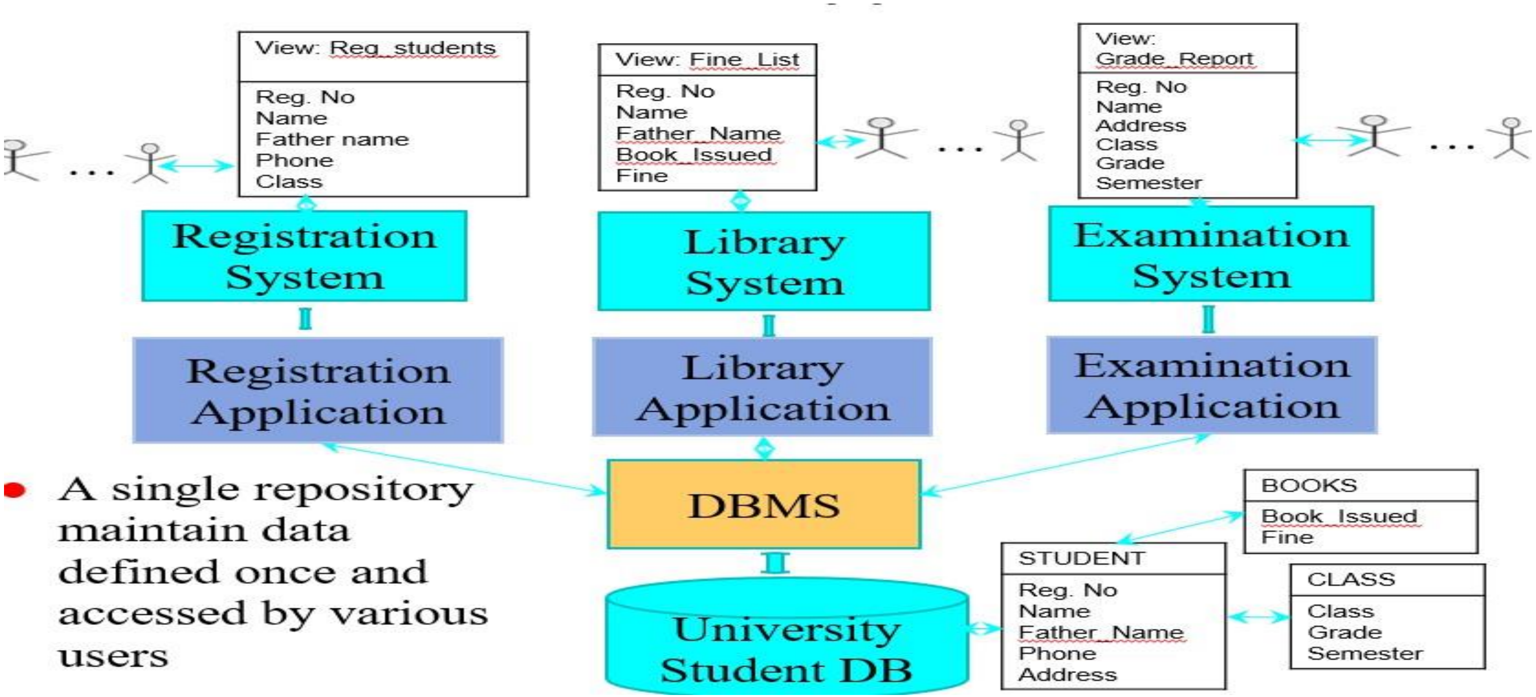
Disadvantages

- Less security
- Data inconsistency
- Data redundancy
- Cumbersome sharing of information
- Slow for huge databases

- **Redundancy** is to store the same data in different files for no need. Redundancy caused TFPs to ***waste storage*** space, to require duplication of effort for multiple updates (***waste time***), and to perhaps show inconsistent data.

- **Inconsistency** is to show different values for the same data item in different places.

Database Approach



- A single repository maintain data defined once and accessed by various users

Databases

A database (db) is an organized collection of data, typically stored in electronic format

- It allows you to input, manage, organize, and retrieve data quickly.
- Traditional databases are organized by records (rows), fields (columns) stored in tables which are stored in the database files.

Relational Databases

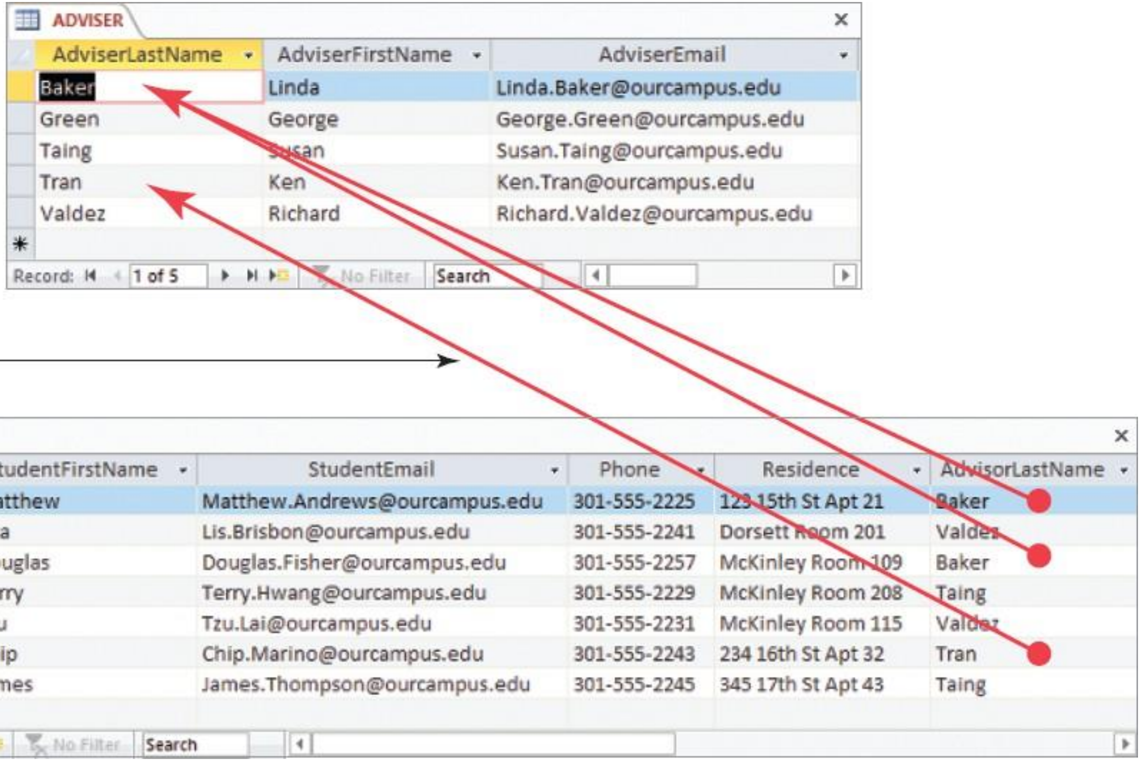
- A relational database stores information in tables.
- The various software systems used to maintain relational databases are known as a relational database management system (RDBMS). Virtually all relational database systems use SQL (Structured Query Language) as the language for querying and maintaining the database.
- A relational informational topic is stored in its own table.
- In essence, a relational database will break-up a list into several parts—one
  part for each theme in the list.
- A Project List would be divided into a CUSTOMER Table, a PROJECT Table, and a PROJECT_MANAGER Table.

Putting the Pieces Back Together

- In our relational database, we broke our list into several tables. Somehow the tables must be joined back together.
- In a relational database, tables are joined together using the value of the data.
- If a PROJECT has a CUSTOMER, the Customer_ID is stored as a column in the PROJECT table. The value stored in this column can be used to retrieve specific customer information from the CUSTOMER table.
- A relational database is more complicated than a list.
- However, a relational database minimizes data redundancy, avoids list modification issues and preserves complex relationships among topics, and allows for partial data.

Relational Database Example

STUDENT data linked to ADVISER data via **AdviserLastName**



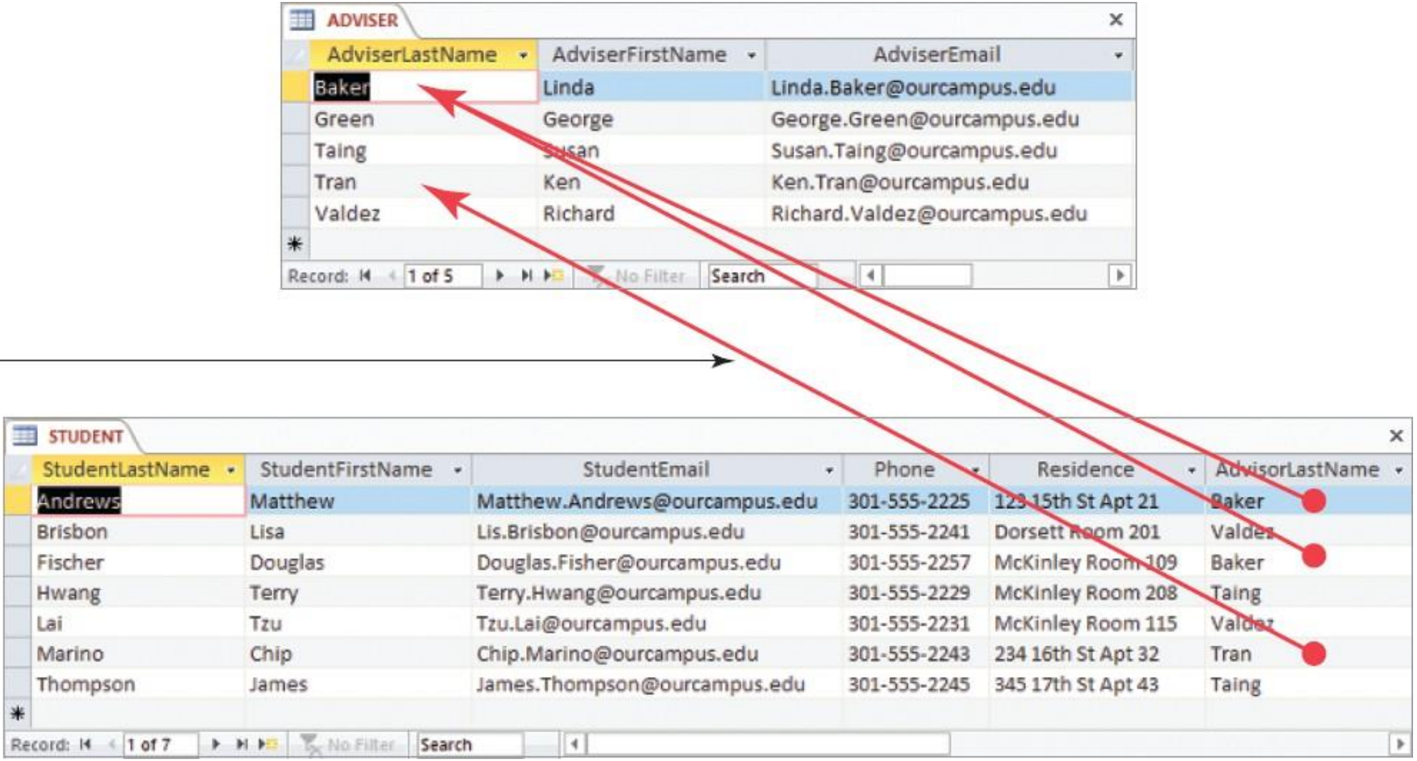
ADVISER

AdviserLastName	AdviserFirstName	AdviserEmail
Baker	Linda	Linda.Baker@ourcampus.edu
Green	George	George.Green@ourcampus.edu
Taing	Susan	Susan.Taing@ourcampus.edu
Tran	Ken	Ken.Tran@ourcampus.edu
Valdez	Richard	Richard.Valdez@ourcampus.edu

STUDENT

StudentLastName	StudentFirstName	StudentEmail	Phone	Residence	AdvisorLastName
Andrews	Matthew	Matthew.Andrews@ourcampus.edu	301-555-2225	123 15th St Apt 21	Baker
Brisbon	Lisa	Lis.Brisbon@ourcampus.edu	301-555-2241	Dorsett Room 201	Valdez
Fischer	Douglas	Douglas.Fisher@ourcampus.edu	301-555-2257	McKinley Room 109	Baker
Hwang	Terry	Terry.Hwang@ourcampus.edu	301-555-2229	McKinley Room 208	Taing
Lai	Tzu	Tzu.Lai@ourcampus.edu	301-555-2231	McKinley Room 115	Valdez
Marino	Chip	Chip.Marino@ourcampus.edu	301-555-2243	234 16th St Apt 32	Tran
Thompson	James	James.Thompson@ourcampus.edu	301-555-2245	345 17th St Apt 43	Taing

Relational Database Example



ADVISER

AdviserLastName	AdviserFirstName	AdviserEmail
Baker	Linda	Linda.Baker@ourcampus.edu
Green	George	George.Green@ourcampus.edu
Taing	Susan	Susan.Taing@ourcampus.edu
Tran	Ken	Ken.Tran@ourcampus.edu
Valdez	Richard	Richard.Valdez@ourcampus.edu

Record: 1 of 5 No Filter Search

STUDENT

StudentLastName	StudentFirstName	StudentEmail	Phone	Residence	AdvisorLastName
Andrews	Matthew	Matthew.Andrews@ourcampus.edu	301-555-2225	123 15th St Apt 21	Baker
Brisbon	Lisa	Lis.Brisbon@ourcampus.edu	301-555-2241	Dorsett Room 201	Valdez
Fischer	Douglas	Douglas.Fisher@ourcampus.edu	301-555-2257	McKinley Room 109	Baker
Hwang	Terry	Terry.Hwang@ourcampus.edu	301-555-2229	McKinley Room 208	Taing
Lai	Tzu	Tzu.Lai@ourcampus.edu	301-555-2231	McKinley Room 115	Valdez
Marino	Chip	Chip.Marino@ourcampus.edu	301-555-2243	234 16th St Apt 32	Tran
Thompson	James	James.Thompson@ourcampus.edu	301-555-2245	345 17th St Apt 43	Taing

Record: 1 of 7 No Filter Search

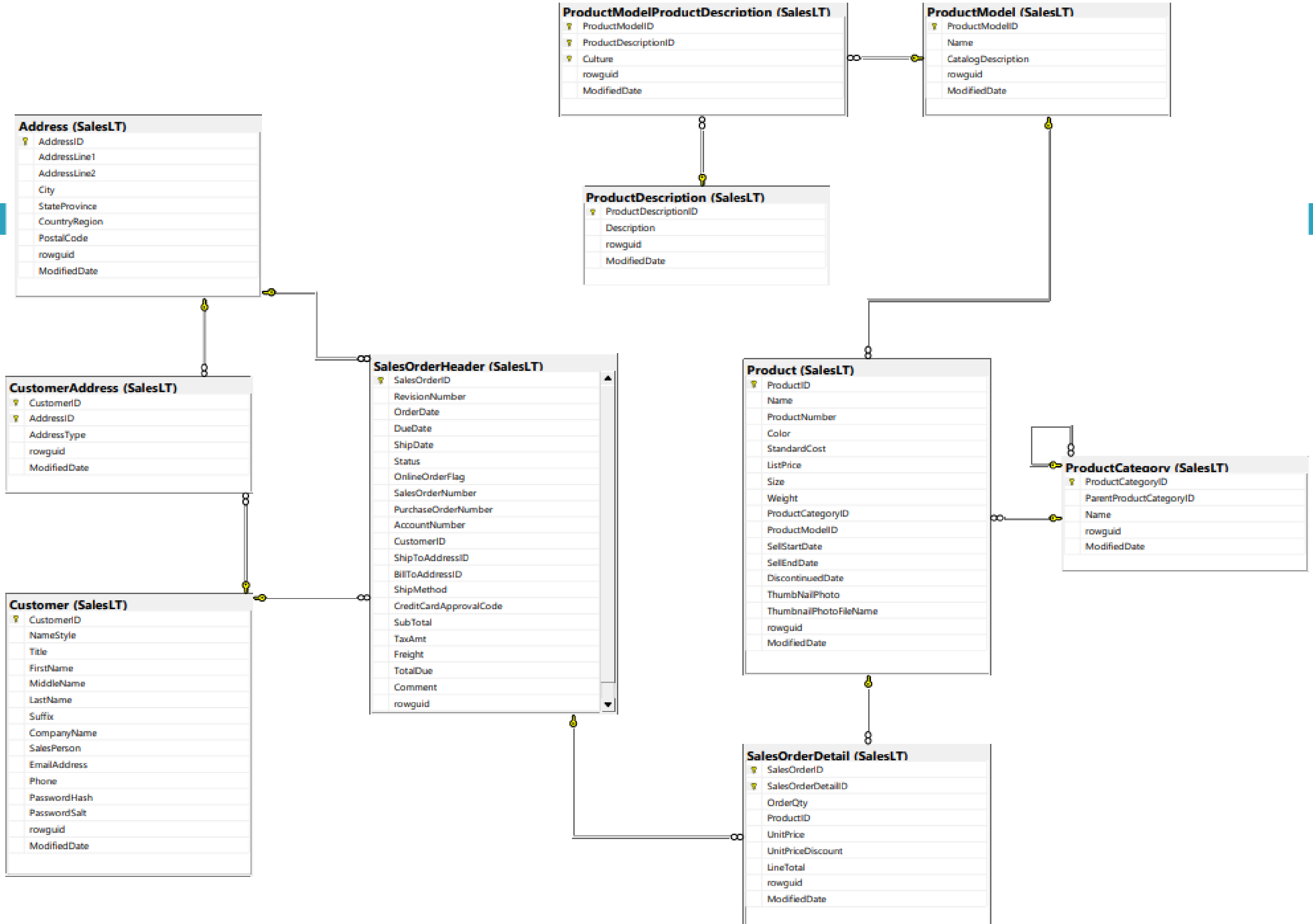
STUDENT data linked to ADVISER data via **AdviserLastName**

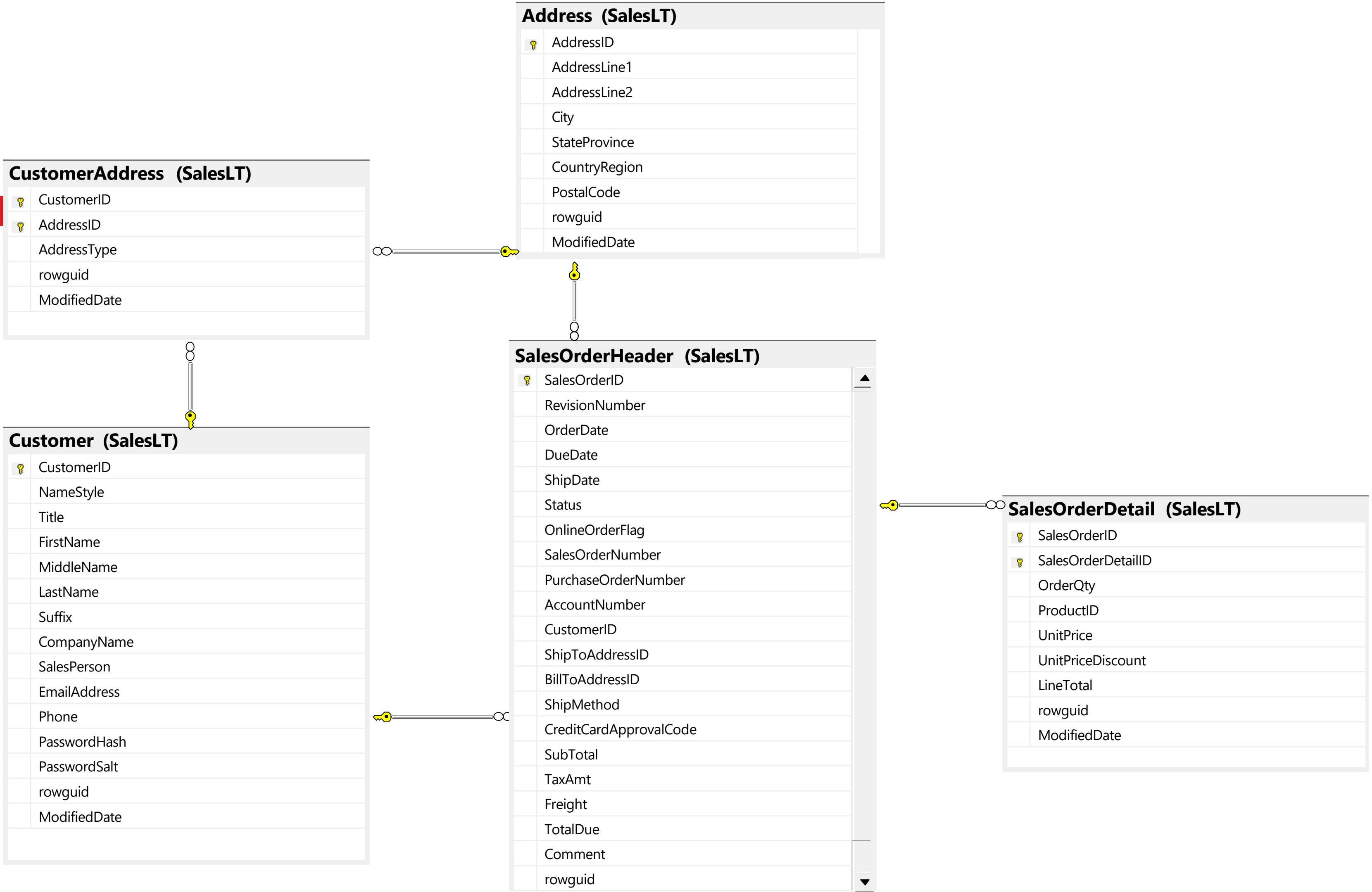
Primary Keys

- The Primary Key uniquely identifies each row in a table
- A table can only have one Primary Key – but this Primary Key can consist of either one column or multiple columns
- The Primary Key must contain unique values
- The Primary Key cannot contain NULL values

Foreign Keys

- A Foreign Key is a column (or collection of columns) that refers to the Primary Key in another table
- The table contain the Foreign Key is called the child table, and the table being referenced is the parent table
- A table can have more than one Foreign Key referring to one or more other tables
- It is possible for a Foreign Key field to allow NULL values
- A Foreign Key field may have a UNIQUE constraint. If so there is then a one-to-one relationship between the two tables
- If there is no UNIQUE constraint, there is a one-to-many relationship between the two tables


ProductModelProductDescription (SalesLT)
ProductModelID
ProductDescriptionID
Culture
rowguid
ModifiedDate
ProductModel (SalesLT)
ProductModelID
Name
CatalogDescription
rowguid
ModifiedDate
Address (SalesLT)
AddressID
AddressLine1
AddressLine2
City
StateProvince
CountryRegion
PostalCode
rowguid
ModifiedDate
ProductDescription (SalesLT)
ProductDescriptionID
Description
rowguid
ModifiedDate
CustomerAddress (SalesLT)
CustomerID
AddressID
AddressType
rowguid
ModifiedDate
SalesOrderHeader (SalesLT)
SalesOrderID
RevisionNumber
OrderDate
DueDate
ShipDate
Status
OnlineOrderFlag
SalesOrderNumber
PurchaseOrderNumber
AccountNumber
CustomerID
ShipToAddressID
BillToAddressID
ShipMethod
CreditCardApprovalCode
SubTotal
TaxAmt
Freight
TotalDue
Comment
rowguid
Product (SalesLT)
ProductID
Name
ProductNumber
Color
StandardCost
ListPrice
Size
Weight
ProductCategoryID
ProductModelID
SellStartDate
SellEndDate
DiscontinuedDate
ThumbNailPhoto
ThumbnailPhotoFileName
rowguid
ModifiedDate
ProductCategory (SalesLT)
ProductCategoryID
ParentProductCategoryID
Name
rowguid
ModifiedDate
Customer (SalesLT)
CustomerID
NameStyle
Title
FirstName
MiddleName
LastName
Suffix
CompanyName
SalesPerson
EmailAddress
Phone
PasswordHash
PasswordSalt
rowguid
ModifiedDate
SalesOrderDetail (SalesLT)
SalesOrderID
SalesOrderDetailID
OrderQty
ProductID
UnitPrice
UnitPriceDiscount
LineTotal
rowguid
ModifiedDate



Address (SalesLT)

- AddressID
- AddressLine1
- AddressLine2
- City
- StateProvince
- CountryRegion
- PostalCode
- rowguid
- ModifiedDate

CustomerAddress (SalesLT)

- CustomerID
- AddressID
- AddressType
- rowguid
- ModifiedDate

Customer (SalesLT)

- CustomerID
- NameStyle
- Title
- FirstName
- MiddleName
- LastName
- Suffix
- CompanyName
- SalesPerson
- EmailAddress
- Phone
- PasswordHash
- PasswordSalt
- rowguid
- ModifiedDate

SalesOrderHeader (SalesLT)

- SalesOrderID
- RevisionNumber
- OrderDate
- DueDate
- ShipDate
- Status
- OnlineOrderFlag
- SalesOrderNumber
- PurchaseOrderNumber
- AccountNumber
- CustomerID
- ShipToAddressID
- BillToAddressID
- ShipMethod
- CreditCardApprovalCode
- SubTotal
- TaxAmt
- Freight
- TotalDue
- Comment
- rowguid

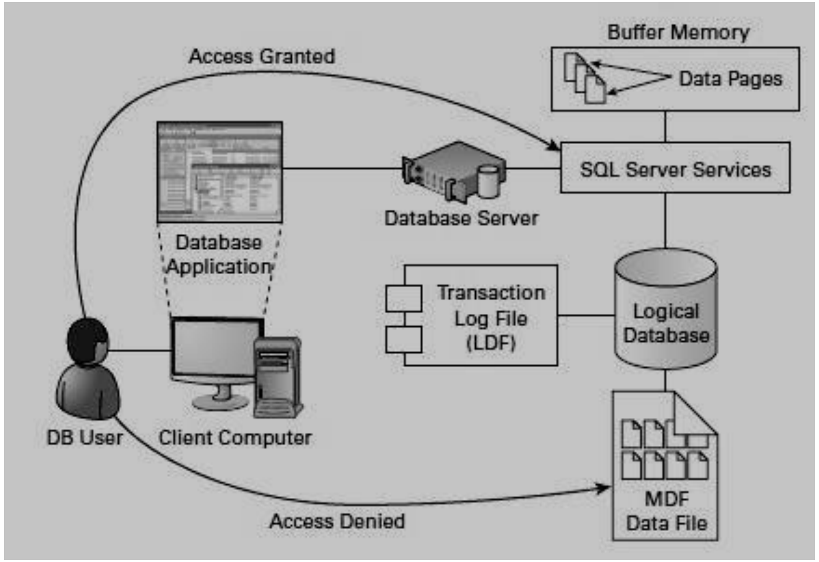
SalesOrderDetail (SalesLT)

- SalesOrderID
- SalesOrderDetailID
- OrderQty
- ProductID
- UnitPrice
- UnitPriceDiscount
- LineTotal
- rowguid
- ModifiedDate

Database Servers

- Databases are stored on database servers which are dedicated physical or virtual servers that host the database files and provide high-level performance for users who are accessing the data.

- Database servers contain the DBMS used to manage the data and administer the SQL Server environment.



Database Management System (DBMS)

- Database Management System (DBMS) is used by the users to access the data stored in database files. A DBMS is also used to perform administrative tasks on the databases and objects contained within the database.
- DBMS is a collection of applications that allows users and other programs to capture and analyze data by providing additional functionality like reporting services to help you create, deploy, and manage reports for your organization.
- A RDBMS is a software system designed to allow the definition, creation, querying, and updating of data stored in relational databases.
- A few examples of RDBMS include; Microsoft SQL Server, Oracle, Microsoft Access, and MySQL.

Most popular database systems

MySQL
ORACLE
Microsoft SQL Server

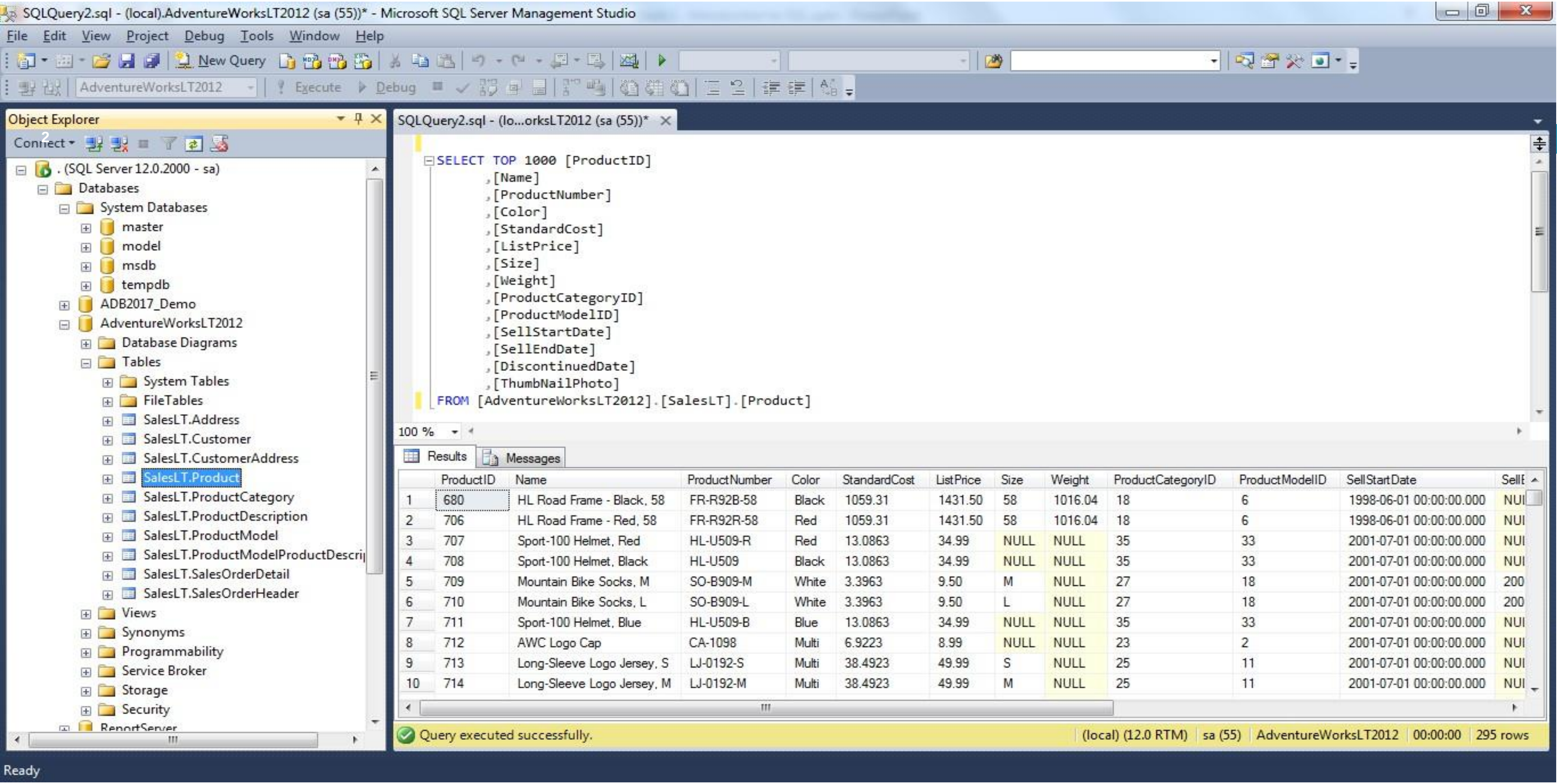
Transact-SQL (T-SQL)

- T-SQL is Microsoft’s implementation of the industry standard Structured Query Language. Originally developed to support the new relational data model at International Business Machines (IBM) in the early 1970s, SQL has gone on to wide adoption in the industry.
- Besides Microsoft’s implementation as T-SQL in SQL Server, Oracle implements SQL as PL/SQL, IBM implements it as SQL PL, and Sybase maintains its own implementation of T- SQL.
- An important concept to understand when working with T-SQL is that it is a set-based and declarative language, not a procedural language.

What Can T-SQL do?

- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database
- SQL can create views in a database
- SQL can set permissions on tables, procedures, and views

SQL Server Management Studio



```sql
SELECT TOP 1000 [ProductID]
      ,[Name]
      ,[ProductNumber]
      ,[Color]
      ,[StandardCost]
      ,[ListPrice]
      ,[Size]
      ,[Weight]
      ,[ProductCategoryID]
      ,[ProductModelID]
      ,[SellStartDate]
      ,[SellEndDate]
      ,[DiscontinuedDate]
      ,[ThumbNailPhoto]
FROM [AdventureWorksLT2012].[SalesLT].[Product]
```

	ProductID	Name	ProductNumber	Color	StandardCost	ListPrice	Size	Weight	ProductCategoryID	ProductModelID	SellStartDate	SellE
1	680	HL Road Frame - Black, 58	FR-R92B-58	Black	1059.31	1431.50	58	1016.04	18	6	1998-06-01 00:00:00.000	NUI
2	706	HL Road Frame - Red, 58	FR-R92R-58	Red	1059.31	1431.50	58	1016.04	18	6	1998-06-01 00:00:00.000	NUI
3	707	Sport-100 Helmet, Red	HL-U509-R	Red	13.0863	34.99	NULL	NULL	35	33	2001-07-01 00:00:00.000	NUI
4	708	Sport-100 Helmet, Black	HL-U509	Black	13.0863	34.99	NULL	NULL	35	33	2001-07-01 00:00:00.000	NUI
5	709	Mountain Bike Socks, M	SO-B909-M	White	3.3963	9.50	M	NULL	27	18	2001-07-01 00:00:00.000	200
6	710	Mountain Bike Socks, L	SO-B909-L	White	3.3963	9.50	L	NULL	27	18	2001-07-01 00:00:00.000	200
7	711	Sport-100 Helmet, Blue	HL-U509-B	Blue	13.0863	34.99	NULL	NULL	35	33	2001-07-01 00:00:00.000	NUI
8	712	AWC Logo Cap	CA-1098	Multi	6.9223	8.99	NULL	NULL	23	2	2001-07-01 00:00:00.000	NUI
9	713	Long-Sleeve Logo Jersey, S	LJ-0192-S	Multi	38.4923	49.99	S	NULL	25	11	2001-07-01 00:00:00.000	NUI
10	714	Long-Sleeve Logo Jersey, M	LJ-0192-M	Multi	38.4923	49.99	M	NULL	25	11	2001-07-01 00:00:00.000	NUI

Query executed successfully. | (local) (12.0 RTM) | sa (55) | AdventureWorksLT2012 | 00:00:00 | 295 rows

What is SQL

- SQL language
  - Standard language for describing relational databases and queries.
  - Considered one of the major reasons for the commercial success of relational databases
- SQL
  - Structured Query Language
  - Statements for data definitions, queries, and updates (both DDL and DML)
  - Core specification
  - Plus specialized extensions

Brief History of SQL

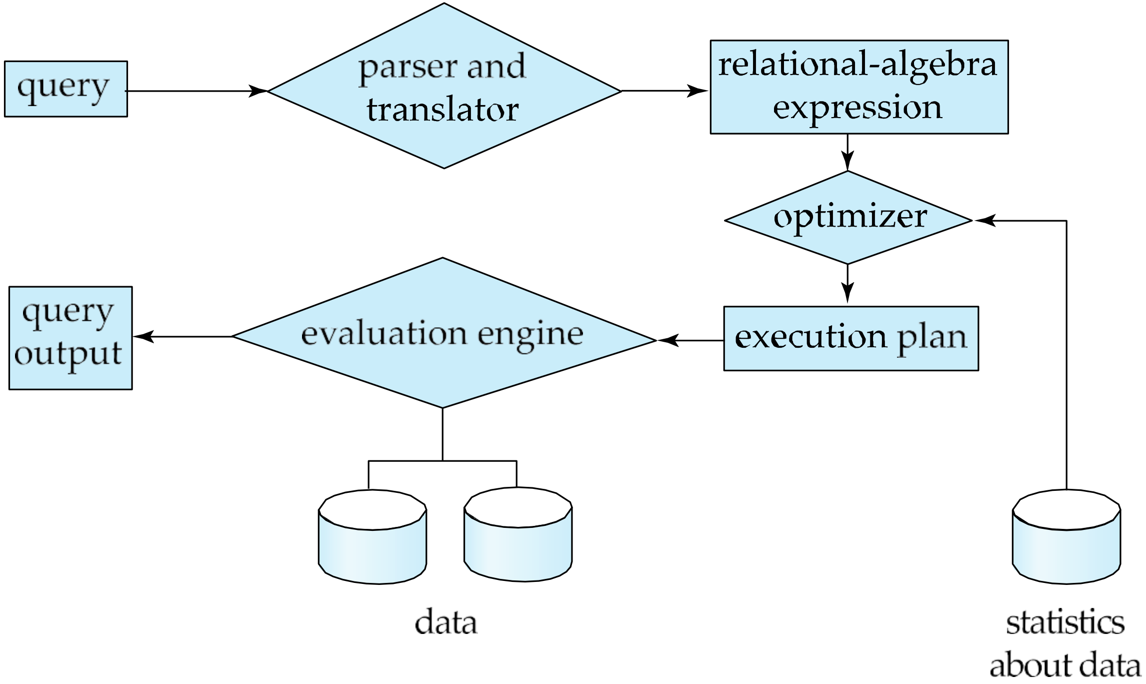
- 1970– E. Codd develops relational database concept
- 1974-1979–System R with Sequel (later SQL) created at IBM Research Lab
- 1979–Oracle markets first relational DB with SQL
- 1986–ANSI SQL standard released
- 1989, 1992, 1999, 2003, 2007–Major ANSI standard updates
- Current–SQL is supported by most major database vendors at 1999, 2003 and 2007

SQL: Language Breakdown

Domain	SQL Commands	Objects
Metadata (DDL) Data Definition language	CREATE ALTER DROP	Tables, functions, views, procedures, etc
DATA (DML) Data Manipulation language	C – INSERT/Create R – SELECT/ Reterieve U - UPDATE D - DELETE	Tables (as a target)
Security (DCL) Data Control language	GRANT REVOKE	Tables, functions, views, procedures, etc.
Transactions (TCL) Transaction Control Language	BEGIN TRANS COMMIT ROLLBACK	Controls DML statements

Processing

1. Parsing and translation
2. Optimization
3. Evaluation



Basic Steps in Query Processing (Cont.)

- **Parsing and translation**

translate the query into its internal form using Parser. This is then translated into relational algebra. Parser perform following activities

- **Verify query is syntactically correct**

verifies relations in query with matching relation in database

- **Construct a parse tree (query –Tree) for representing expression using views**

Convert the parse tree into Relational Algebra

- **Query optimization:**

Selection of an efficient query evaluation plan is made

- **Evaluation**

The query-execution engine takes a query-evaluation plan, executes that plan, and returns the answers to the query.

SQL Statement Types

Data Manipulation Language (DML)	Data Definition Language (DDL)	Data Control Language (DCL)	Transaction Control Language (TCL)
Statements for querying and modifying data: • SELECT • INSERT • UPDATE • DELETE	Statements for defining database objects: • CREATE • ALTER • DROP	Statements for assigning security permissions: • GRANT • REVOKE • DENY	Statements for maintaining database consistency & managing transactions: • COMMIT • ROLLBACK

Data Manipulation Language (DML)

Data Manipulation Language (DML) is a vocabulary used to retrieve and work with data in SQL Server. Use these statements to add, modify, query, or remove data from a SQL Server database.

Common DML statements

- **SELECT – retrieve data**
- **INSERT – add data**
- **UPDATE – modify data**
- **DELETE – remove data**

Data Definition Language (DDL)

Data Definition Language (DML) consists of all the commands that can be used to define the structure of the database. It is used to create or modify database objects – for example, tables, indexes, functions, views, stored procedures and triggers).

Common DDL statements

- **CREATE – used to create the database and database objects**
- **DROP – used to delete objects from the database**
- **ALTER – used to alter the structure of the database**

Data Control Language (DCL)

Data Control Language deals with rights, permissions and other controls of the database system.

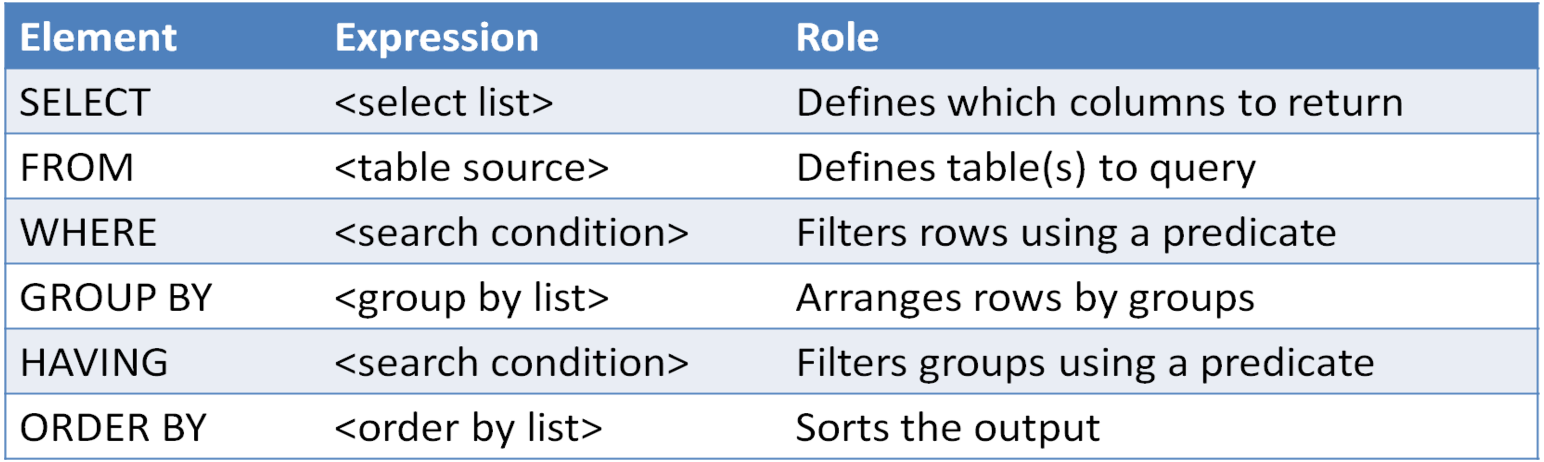
Common DCL statements

- **GRANT – allow specified users to perform specified tasks**
- **REVOKE – take away permissions from a user or group**

DML - SELECT statement

The SELECT statement is used to query tables and views. You may also perform some manipulation of the data with SELECT before returning the results. It is likely that you will use the SELECT statement more than any other single statement in T-SQL. This module introduces you to the fundamentals of the SELECT statement, focusing on queries against a single table.

Elements of the SELECT Statement



Element	Expression	Role
SELECT	<select list>	Defines which columns to return
FROM	<table source>	Defines table(s) to query
WHERE	<search condition>	Filters rows using a predicate
GROUP BY	<group by list>	Arranges rows by groups
HAVING	<search condition>	Filters groups using a predicate
ORDER BY	<order by list>	Sorts the output

Filtering Data with WHERE Clause

In order to limit the rows that are returned by your query, you will need to add a WHERE clause to your SELECT statement, following the FROM clause. WHERE clauses are constructed from a search condition, which in turn is written as a predicate expression. The purpose of the predicate is to provide a logical filter through which each row must pass. Only rows returning TRUE in the predicate will be output to the next logical phase of the query. WHERE clause is logically the next phase in query execution after FROM, so it will be processed before other clauses such as SELECT.

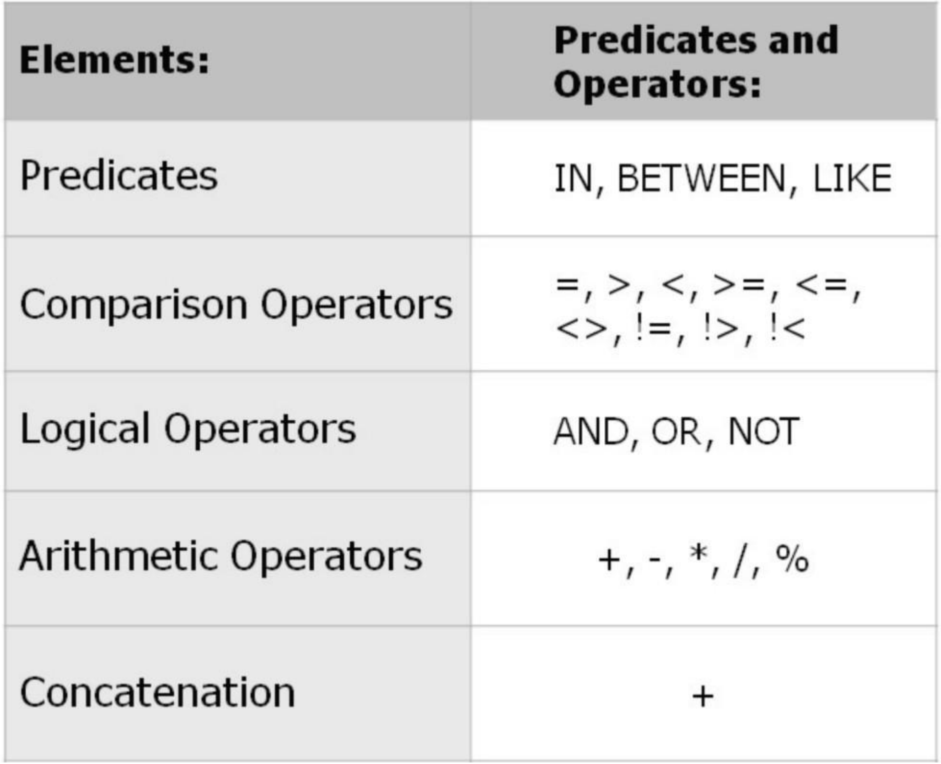
```
SELECT *
FROM   SalesLT.Product
WHERE Color=‘Red’;
```

Filtering Data with WHERE Clause

In order to limit the rows that are returned by your query, you will need to add a WHERE clause to your SELECT statement, following the FROM clause. WHERE clauses are constructed from a search condition, which in turn is written as a predicate expression. The purpose of the predicate is to provide a logical filter through which each row must pass. Only rows returning TRUE in the predicate will be output to the next logical phase of the query. WHERE clause is logically the next phase in query execution after FROM, so it will be processed before other clauses such as SELECT.

```
SELECT *
FROM   SalesLT.Product
WHERE Color=‘Red’;
```

T-SQL Language Elements: Predicates and Operators



Elements:	Predicates and Operators:
Predicates	IN, BETWEEN, LIKE
Comparison Operators	=, >, <, >=, <=, <>, !=, !>, !<
Logical Operators	AND, OR, NOT
Arithmetic Operators	+, -, *, /, %
Concatenation	+

As with other mathematical environments, operators are subject to rules governing precedence. The following list describes the order in which T-SQL operators are evaluated:

Order of Evaluation Operators

1. ( ) Parentheses
2. *, / , % (Multiply, Divide, Modulo)
3. +, - (Add/Positive/Concatenate, Subtract/Negative)
4. =, <, >, >=, <=, !=, !>, !< (Comparison)
5. NOT
6. AND
7. BETWEEN, IN, LIKE, OR
8. = (Assignment)

Filtering Data with WHERE Clause

```
SELECT *
FROM    SalesLT.Product
WHERE Color <> ‘Red’;
```

```
SELECT *
FROM    SalesLT.Product
WHERE Color IN (‘Red’);
```

```
SELECT *
FROM    SalesLT.Product
WHERE Color IN (‘Red’,‘Black’);
```

Wild Card Operators in SQL

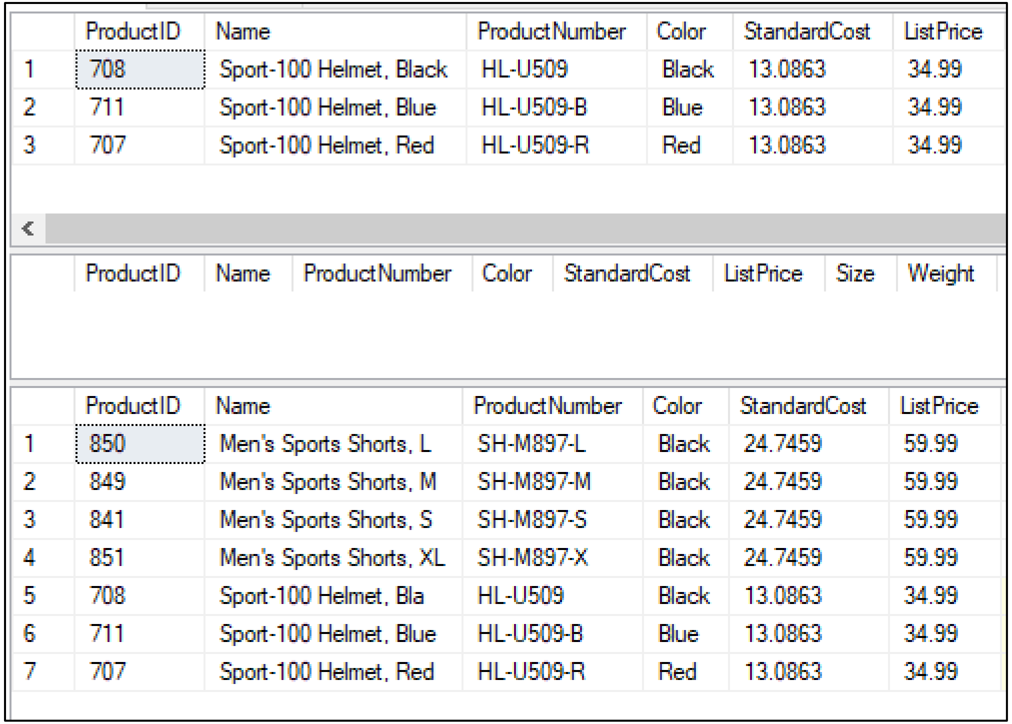
Wildcard character	Description	Example
%	Any string of zero or more characters.	**WHERE title LIKE '%computer%'** finds all book titles with the word 'computer' anywhere in the book title.
_ (underscore)	Any single character.	**WHERE au_fname LIKE '_ean'** finds all four-letter first names that end with ean (Dean, Sean, and so on).
[ ]	Any single character within the specified range ([a-f]) or set ([abcdef]).	**WHERE au_lname LIKE '[C-P]arsen'** finds author last names ending with arsen and starting with any single character between C and P, for example Carsen, Larsen, Karsen, and so on. In range searches, the characters included in the range may vary depending on the sorting rules of the collation.
[^]	Any single character not within the specified range ([^a-f]) or set ([^abcdef]).	**WHERE au_lname LIKE 'de[^l]%'** all author last names starting with de and where the following letter is not l.

Filtering Data with Wildcard Character %

```
SELECT *
FROM [SalesLT].[Product]
WHERE Name Like 'Sport%'
```

```
SELECT *
FROM [SalesLT].[Product]
WHERE Name Like ‘%Sport'
```

```
SELECT *
FROM [SalesLT].[Product]
WHERE Name Like ‘%Sport%'
```



	ProductID	Name	ProductNumber	Color	StandardCost	ListPrice
1	708	Sport-100 Helmet, Black	HL-U509	Black	13.0863	34.99
2	711	Sport-100 Helmet, Blue	HL-U509-B	Blue	13.0863	34.99
3	707	Sport-100 Helmet, Red	HL-U509-R	Red	13.0863	34.99

	ProductID	Name	ProductNumber	Color	StandardCost	ListPrice	Size	Weight

	ProductID	Name	ProductNumber	Color	StandardCost	ListPrice
1	850	Men's Sports Shorts, L	SH-M897-L	Black	24.7459	59.99
2	849	Men's Sports Shorts, M	SH-M897-M	Black	24.7459	59.99
3	841	Men's Sports Shorts, S	SH-M897-S	Black	24.7459	59.99
4	851	Men's Sports Shorts, XL	SH-M897-X	Black	24.7459	59.99
5	708	Sport-100 Helmet, Bla	HL-U509	Black	13.0863	34.99
6	711	Sport-100 Helmet, Blue	HL-U509-B	Blue	13.0863	34.99
7	707	Sport-100 Helmet, Red	HL-U509-R	Red	13.0863	34.99

Removing Duplicates

- **SELECT ALL**

Default behavior includes duplicates

```
SELECT Color
FROM SalesLT.Product
```

Color
Blue
Red
Yellow
Blue
Yellow
Black

- **SELECT DISTINCT**

Removes duplicates

```
SELECT DISTINCT Color
FROM SalesLT.Product
```

Color
Blue
Red
Yellow
Black

Sorting Results

- □ Use ORDER BY to sort results by one or more columns

  You can specify ASC or DESC (ASC is the default)

  You can order by columns in the source that are not included in the SELECT clause

  Aliases created in SELECT clause are visible to ORDER BY

```
SELECT *
FROM SalesLT.Product
ORDER BY Name
```

```
SELECT *
FROM SalesLT.Product
ORDER BY ProductNumber
```

```
SELECT *
FROM SalesLT.Product
ORDER BY ListPrice DESC;
```

Limiting Sorted Results

- TOP allows you to limit the number or percentage of rows returned by a query
- Works with ORDER BY clause to limit rows by sort order
- Added to SELECT clause:

SELECT TOP (N)

```
SELECT *
FROM SalesLT.Product
ORDER BY ListPrice DESC;
```

```
SELECT TOP 10  *
FROM SalesLT.Product
ORDER BY ListPrice DESC;
```

```
SELECT TOP 10 PERCENT *
FROM SalesLT.Product
ORDER BY ListPrice DESC;
```

What is a NULL value?

A field with a NULL value is one that was left blank when the record was created. It is different to a zero value or a field containing spaces.

In SQL, a NULL value can be thought of as meaning that the actual value could be anything, so you cannot use comparison operators like = or <> with NULL.

The below statement will not work!

```
SELECT *
FROM   SalesLT.Product
WHERE Color = NULL;
```

Checking for NULL values

To check NULL values, we have two special keywords we can use

```
IS NULL
IS NOT NULL
```

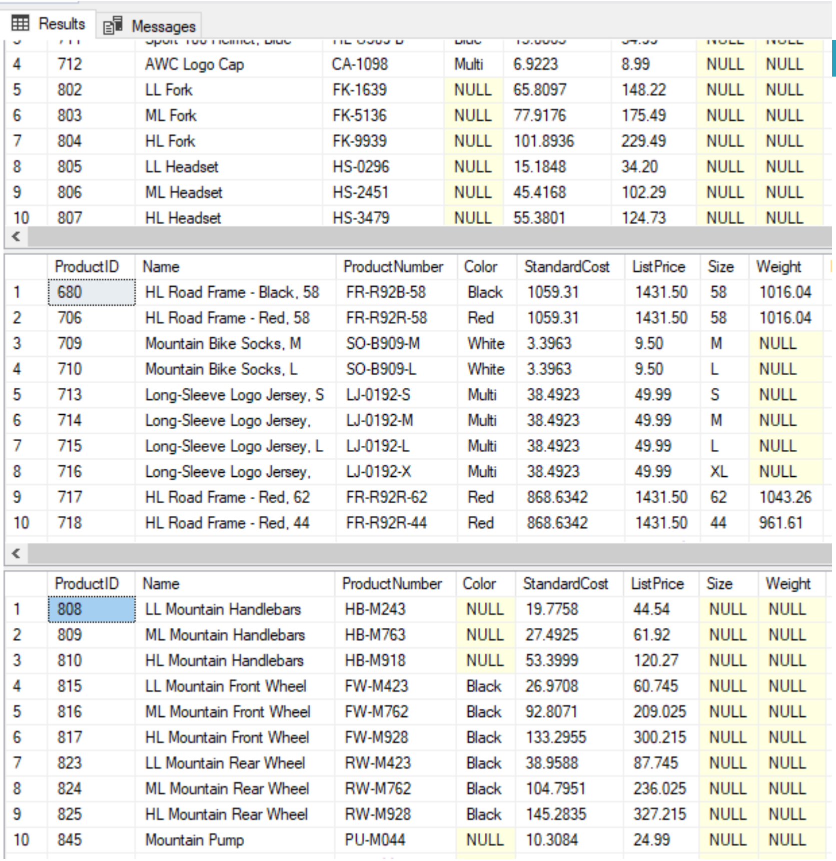
```
SELECT *
FROM   SalesLT.Product
WHERE Color IS NULL;
```

Working with NULL Values

```
SELECT *
FROM [SalesLT].[Product]
WHERE Size IS NULL

SELECT *
FROM [SalesLT].[Product]
WHERE Size IS NOT NULL

SELECT *
FROM [SalesLT].[Product]
WHERE Name LIKE '%Mountain%'
          AND Size IS NULL
```



4	712	AWC Logo Cap	CA-1098	Multi	6.9223	8.99	NULL	NULL
5	802	LL Fork	FK-1639	NULL	65.8097	148.22	NULL	NULL
6	803	ML Fork	FK-5136	NULL	77.9176	175.49	NULL	NULL
7	804	HL Fork	FK-9939	NULL	101.8936	229.49	NULL	NULL
8	805	LL Headset	HS-0296	NULL	15.1848	34.20	NULL	NULL
9	806	ML Headset	HS-2451	NULL	45.4168	102.29	NULL	NULL
10	807	HL Headset	HS-3479	NULL	55.3801	124.73	NULL	NULL

	ProductID	Name	ProductNumber	Color	StandardCost	ListPrice	Size	Weight
1	680	HL Road Frame - Black, 58	FR-R92B-58	Black	1059.31	1431.50	58	1016.04
2	706	HL Road Frame - Red, 58	FR-R92R-58	Red	1059.31	1431.50	58	1016.04
3	709	Mountain Bike Socks, M	SO-B909-M	White	3.3963	9.50	M	NULL
4	710	Mountain Bike Socks, L	SO-B909-L	White	3.3963	9.50	L	NULL
5	713	Long-Sleeve Logo Jersey, S	LJ-0192-S	Multi	38.4923	49.99	S	NULL
6	714	Long-Sleeve Logo Jersey,	LJ-0192-M	Multi	38.4923	49.99	M	NULL
7	715	Long-Sleeve Logo Jersey, L	LJ-0192-L	Multi	38.4923	49.99	L	NULL
8	716	Long-Sleeve Logo Jersey,	LJ-0192-X	Multi	38.4923	49.99	XL	NULL
9	717	HL Road Frame - Red, 62	FR-R92R-62	Red	868.6342	1431.50	62	1043.26
10	718	HL Road Frame - Red, 44	FR-R92R-44	Red	868.6342	1431.50	44	961.61

	ProductID	Name	ProductNumber	Color	StandardCost	ListPrice	Size	Weight
1	808	LL Mountain Handlebars	HB-M243	NULL	19.7758	44.54	NULL	NULL
2	809	ML Mountain Handlebars	HB-M763	NULL	27.4925	61.92	NULL	NULL
3	810	HL Mountain Handlebars	HB-M918	NULL	53.3999	120.27	NULL	NULL
4	815	LL Mountain Front Wheel	FW-M423	Black	26.9708	60.745	NULL	NULL
5	816	ML Mountain Front Wheel	FW-M762	Black	92.8071	209.025	NULL	NULL
6	817	HL Mountain Front Wheel	FW-M928	Black	133.2955	300.215	NULL	NULL
7	823	LL Mountain Rear Wheel	RW-M423	Black	38.9588	87.745	NULL	NULL
8	824	ML Mountain Rear Wheel	RW-M762	Black	104.7951	236.025	NULL	NULL
9	825	HL Mountain Rear Wheel	RW-M928	Black	145.2835	327.215	NULL	NULL
10	845	Mountain Pump	PU-M044	NULL	10.3084	24.99	NULL	NULL

What we’ve covered…

- Reasons for using a database
- How using related tables helps you avoid the problems of using lists
- The purpose of the database management system (DBMS)
- Introduction to SQL
- Introduction to SELECT queries
- Using predicates, operators and wildcards in queries
- Using DISTINCT, ORDER BY and TOP in SELECT queries
- Working with NULL values