Course Contents

Index

Overview

1. Understanding Database Concepts
2. Planning and Designing Databases
3. Understanding Data Storage
4. Introducing Microsoft SQL Server
5. Creating Database Objects
6. Manipulating Data
7. Administrating Databases

Understanding Database Concepts

Module 1

Overview

1. Database Concepts
2. Database Systems
3. Demo
4. Summary

Database Concepts

- Database
- Data and Information
- Early Database Models
- Contemporary Database Models

Database

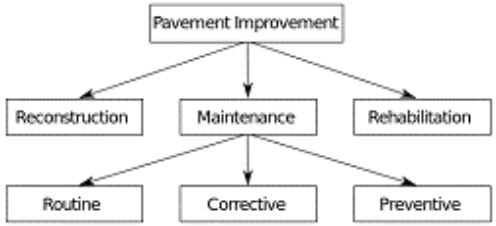
- A database (db) is:
  - An organized collection of related data
  - Is used for easy access and data manipulation
  - Typically stored in electronic format
- Steps in manipulating data:
  - Input data > Organize data > Retrieve data
- Traditional databases are organized by:
  - Files
  - Tables
  - Fields
  - Records

Data and Information

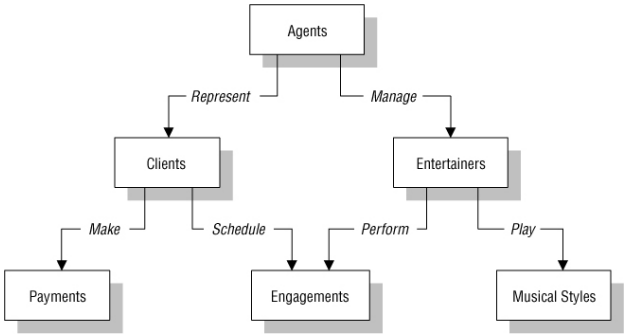
- Data
  - Raw facts which may be numbers, values, names, dates, etc.
- Related data
  - Data which belong to the same entity (person, place, event, etc.)
- Information
  - When process related data it gives some information (doctors name, students' attendance data etc..)
  - Information is useful to take decisions

Early Database Models

- A Hierarchical Database design is similar to a tree structure
  - In a family tree, each parent can have multiple children, but each child can have only one parent

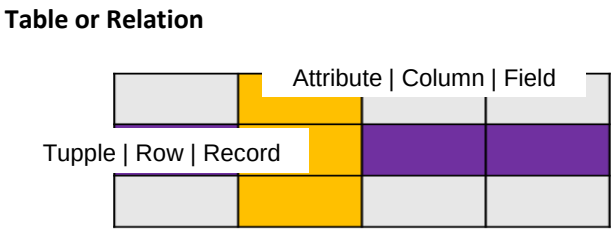


- In Network Database Model the structure of a network database is represented in terms of nodes and set structures

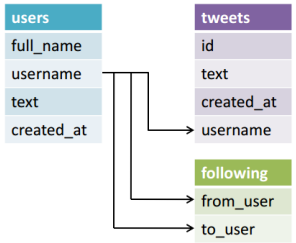


Contemporary Database Models

- Relational database
  - Since the 1970s, relational databases have dominated the field of database systems
  - Codd, Edgar Frank, considered data could be organised into relations consisting of tupples, each with consistent attributes

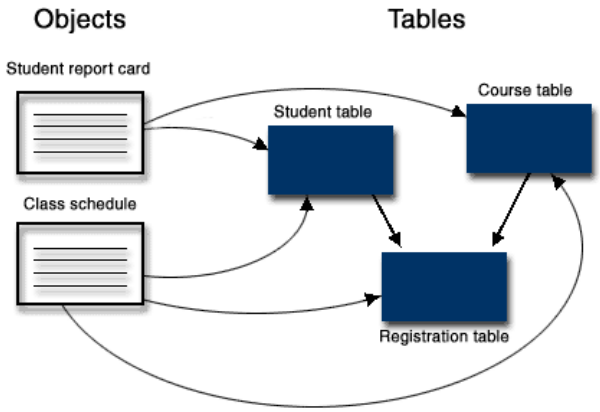


  - The data is stored in tables and any new information is automatically added into the table without the need to reorganize the table itself



Contemporary Database Models

- **Object database**
  - Aka O-Data or object-oriented database management system, OODBMS
  - Is a database management system in which information is represented in the form of objects, as used in object-oriented programming
- Used by:
  - CAD/CAM systems
  - CASE tools
  - Geographic information systems (GIS)
  - Document storage and retrieval systems
  - Artificial intelligence (AI) systems and expert systems



Database Systems

- Database System
- Database System Objectives
- Database System Components
- Database Management System
- DBMSs Examples

Database System

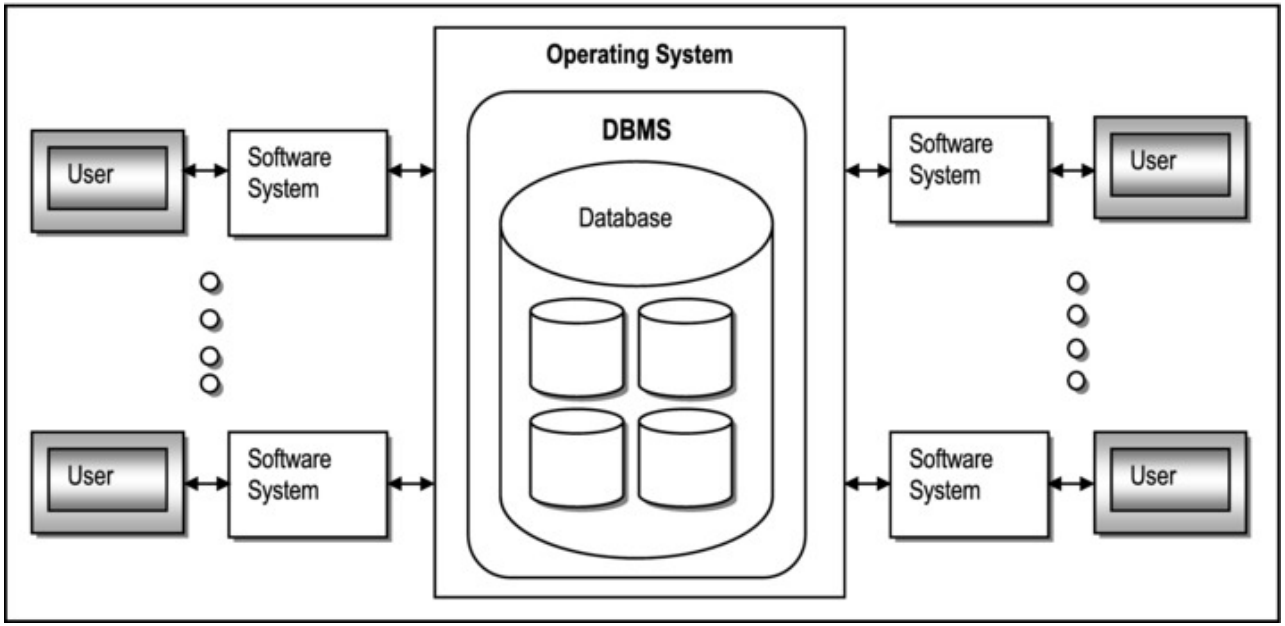
- Past: manual record keeping system
- Recent Past, Present and Future: computerized record keeping system with overall purpose of:
  - Maintaining information
  - Making it available whenever required

Database System Objectives

- Security from unauthorized users and/or systems
- Reliability of performance
- Multiuser support
- Ease and flexibility of data access and data manipulation
- Accuracy and consistency
- Clarity by adopting standards of data representation
- Minimization of data proliferation
- Availability of data whenever it is required
- Physical data independence
- Logical data independence
- Control of redundancy
- Referential integrity
- Tunability

Database System Components

- Hardware and operating system
- Database management system (DBMS) and databases
- Related software systems and/or applications
- End users



Database Management System

- DBMS
  - Stands for Database Management System
  - Is the software that facilitates management and access to the database
  - When a user issues a request via some DSL, for example T-SQL, it's the DBMS that interprets the request, executes the appropriate instructions and responds to the user
- RDBMS
  - Stands for Relational Database Management System
  - Ex. Microsoft SQL Server
- SQL
  - Stands for Strucured Query Language
  - Is a domain-specific language standard, used for managing data held in a RDBMS

DBMSs Examples

- Some examples:

Manufacturer	Product	Type	License
Microsoft	SQL Server	Relational	Paid
Microsoft	Access	Relational	Paid
Oracle	Database	Relational	Paid
Oracle	MySQL	Relational	Freeware
IBM	DB2	Relational	Paid
PostgreSQL	PostgreSQL License	Object-Relational	Freeware
MongoDB	MongoDB	NoSQL	Freeware

Demo

- Explore Hyper-V environment if used, otherwise ignore
- Explore Microsoft SQL Server

Summary

- A database (db) is an organized collection of data, typically stored in electronic format. It allows you to input data, organize the data and retrieve the data quickly.
- A relational database is similar to a hierarchical database in that data is stored in tables and any new information is automatically added into the table without the need to reorganize the table itself. Different from hierarchical database, a table in a relational database can have multiple parents.
- Databases are often found on database servers so that they can be accessed by multiple users and to provide a high-level of performance.

Planning and Designing Databases

Module 2

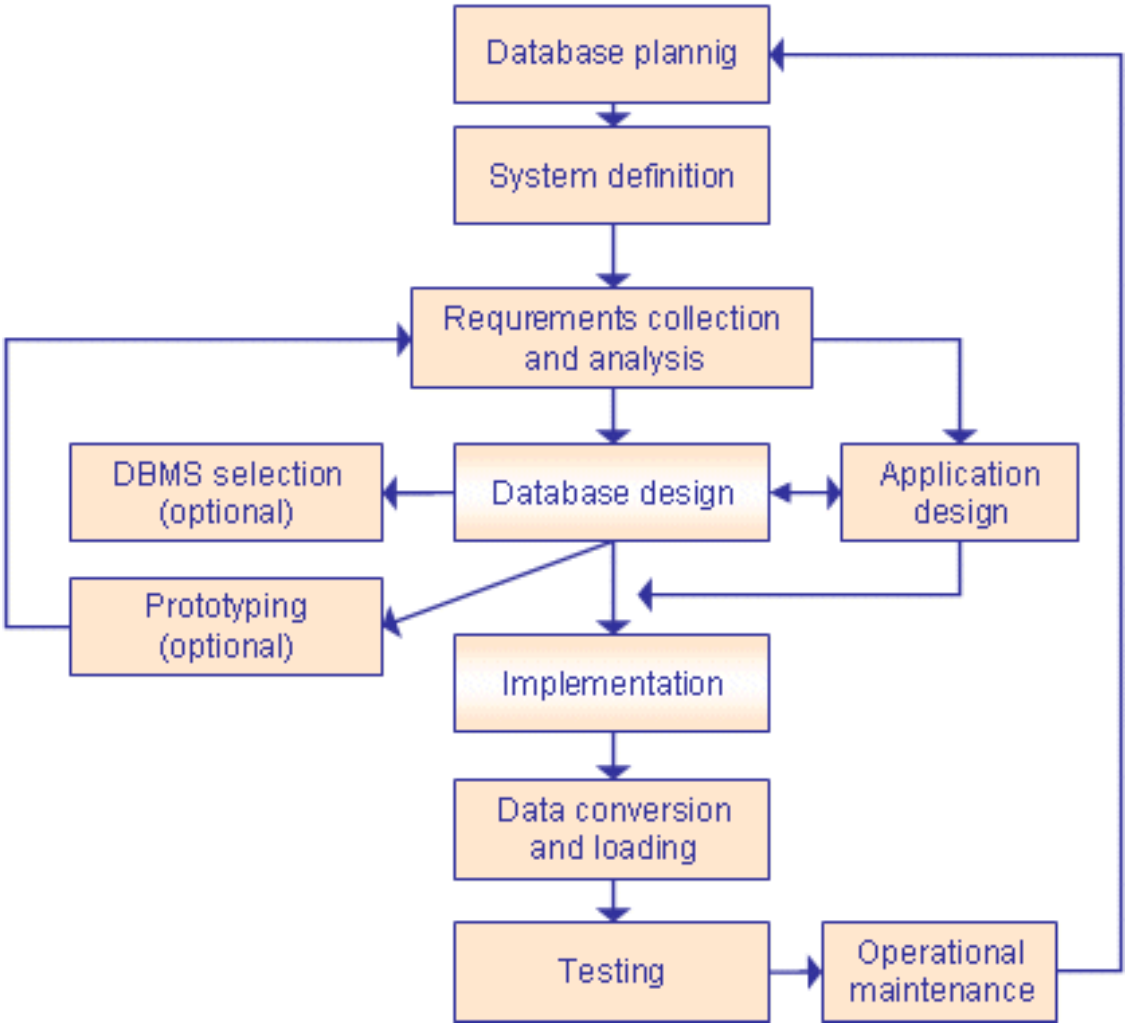
Overview

1. Planning Databases
2. Designing Databases
3. Database Design Technics
4. Demo
5. Practice
6. Summary

Planning Databases

- Database Development Lifecycle
- Database Planning
- Requirements Collection and Analysis
- Database System Example

Database Development Lifecycle



Database Planning

- The database-planning phase begins when a customer requests to develop a database project
- Phases:
  - Decide the tasks
  - Decide the resources
- Major activities:
  - Review and approve the database project request
  - Prioritize the database project request
  - Allocate resources such as money, people and tools
  - Arrange a development team to develop the database project
- Include standards that govern: data, format, documentation

Requirements Collection and Analysis

- Requirements gathering:
  - information-finding is used throughout the database application lifecycle.
  - Enables developer to learn about the:
    - Terminology > Problems > Opportunities > Constraints > Requirements > Priorities of the organization and the users of the system
  - Anwers:
    - What data is needed? How will the data be used?
    - What rules govern the use of the data?
    - Will the data need to be integrated with other systems?
    - What type of a database is required?
    - Where is the data now?
  - The facts are captured using fact-finding techniques:
    - Interviews > Examining documentation > Observing the organization in operation > Research > Questionnaires > Prototyping > Brainstorming

Requirements Collection and Analysis

- Requirements analysis:
  - To delineate the data requirements of the enterprise in terms of basic data elements
  - To describe the information about the data elements and the relationships among them needed to model these data requirements
  - To determine the types of transactions that are intended to be executed on the database and the interaction between the transactions and the data elements
  - To define any performance, integrity, security, or administrative constraints that must be imposed on the resulting database
  - To specify any design and implementation constraints, such as specific technologies, hardware and software, policies, standards, or external interfaces
  - To thoroughly document all of the preceding in a detailed requirements specification

Example
DreamHome Database System

- DreamHome Database System
- Mission statement:
  - The purpose of the DreamHome database system is to maintain the data that is used and generated to support the property rentals for our clients and property owners and to facilitate the cooperation and sharing of information between branches
- Mission objectives:
  - To maintain data (insert, update, delete) on: branches, staff, properties, owners, clients, viewings, leases and advertising
  - To perform searches on: branches, staff, properties, owners, clients, viewings, leases and advertising
  - To track the status of: property for renting, clients wishing to rent and leases
  - To report on: branches, staff, properties, owners, clients, viewings, leases and advertising

Example
DreamHome Database System

- Information-finding techniques

DreamHome
Staff Listing

Branch Number B003

Branch Address
163 Main St, Glasgow
G11 9QX

Telephone Number(s)
0141-339-2178 / 0141-339-4439

Staff Number	Name	Position
SG5	Susan Brand	Manager
SG14	David Ford	Supervisor
SG37	Ann Beech	Assistant
SG112	Annet Longhorn	Supervisor
SG126	Chris Lawrence	Assistant
SG132	Sofie Walters	Assistant

Example
DreamHome Database System

- Information-finding techniques

DreamHome
Property Registration Form

Property Number PG16

Type Flat **Rooms** 4

Rent 450

Address
5 Novar Drive,
Glasgow, G12 9AX

Owner Number C093
(If known)

Person/Business Name
Tony Shaw

Address 12 Park Pl,
Glasgow G4 0QR

Tel No 0141-225-7025

Enter details where applicable

Type of business _____

Contact Name _____

Managed by staff
David Ford

Registered at branch
163 Main St, Glasgow

Example
DreamHome Database System

- Information-finding techniques

DreamHome
Client Registration Form

Client Number CR74
(Enter if known)

Full Name
Mike Ritchie

Branch Number B003

Branch Address
163 Main St, Glasgow

Enter property requirements

Type Flat

Max Rent 750

Registered By
Ann Beech

Date Registered 16-Nov-02

Example
DreamHome Database System

- Information-finding techniques

DreamHome
Property Listing for Week beginning 01/06/04

If you are interested in viewing or renting any of the properties in this l st please contact our branch office as soon as poss ble.

Branch Address

163 Main St, Glasgow

G11 9QX

Telephone Number(s)

0141-339-2178 / 0141-339-4439

Property No	Address	Type	Rooms	Rent
PG4	6 Lawrence St, Glasgow	Flat	3	350
PG36	2 Manor Rd, Glasgow	Flat	3	375
PG21	18 Dale Road, Glasgow	House	5	600
PG16	5 Novar Drive, Glasgow	Flat	4	450
PG77	100A Apple Lane, Glasgow	House	6	560
PG81	781 Greentree Dr, Glasgow	Flat	4	440

Example
DreamHome Database System

- Information-finding techniques

DreamHome
Property Viewing Report

Property Numner PG4

Type Flat

Rent 350

Property Address
6 Lawrence St, Glasgow

Client No	Name	Date	Comments
CR76	John Kay	20/04/04	Too remote.
CR56	Aline Stewart	26/05/04	
CR74	Mike Ritchie	11/11/04	
CR62	Mary Tregear	11/11/04	OK, but needs redecoration throughout.

Example
DreamHome Database System

- Information-finding techniques

***DreamHome* Lease**
Number 00345810

Client Number CR74 **(Enter if known)** **Full Name** Mike Ritchie **(Please print)** **Client Signature** _____	**Property Number** PG16 **Property Address** 5 Novar Dr, Glasgow
Enter payment details **Monthly Rent** 450 **Payment Method** Cheque **Deposit Paid (Y or N)** Yes	**Rent Start** 01/06/04 **Rent Finish** 31/05/05 **Duration** 1 year

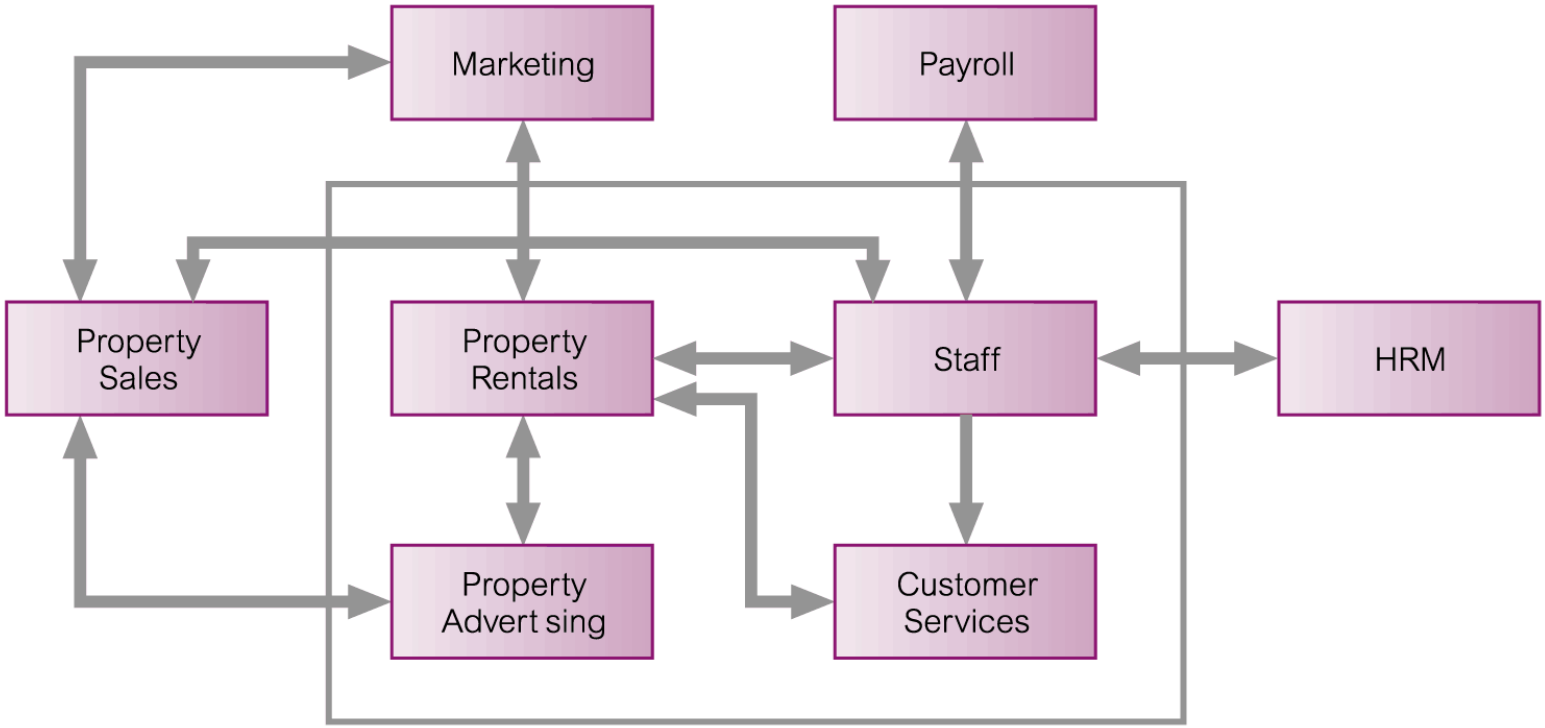
Example
DreamHome Database System

- Major user views

Data	Access Type	Director	Manager	Supervisor	Assistant
All Branches	Maintain				
	Query	X	X		
	Report	X	X		
Single Branch	Maintain		X		
	Query		X		
	Report		X		
All Staff	Maintain				
	Query	X	X		
	Report	X	X		
Branch Staff	Maintain		X		
	Query		X	X	
	Report		X	X	
All Property	Maintain				
	Query	X			
	Report	X	X		
Branch Property	Maintain		X	X	
	Query		X	X	X
	Report		X	X	X
All Owners	Maintain				
	Query	X			
	Report	X	X		
Branch Owners	Maintain		X	X	
	Query		X	X	X
	Report		X		
All Clients	Maintain				
	Query	X			
	Report	X	X		
Branch Clients	Maintain		X	X	
	Query		X	X	X
	Report		X		
All Viewings	Maintain				
	Query				
	Report				
Branch Viewings	Maintain			X	X
	Query			X	X
	Report			X	X
All Leases	Maintain				
	Query	X			
	Report	X	X		
Branch Leases	Maintain		X	X	
	Query		X	X	X
	Report		X	X	
All Newspapers	Maintain				
	Query	X			
	Report	X	X		
Branch Newspapers	Maintain		X		
	Query		X		
	Report		X		

Example
DreamHome Database System

- System boundary



Designing Databases

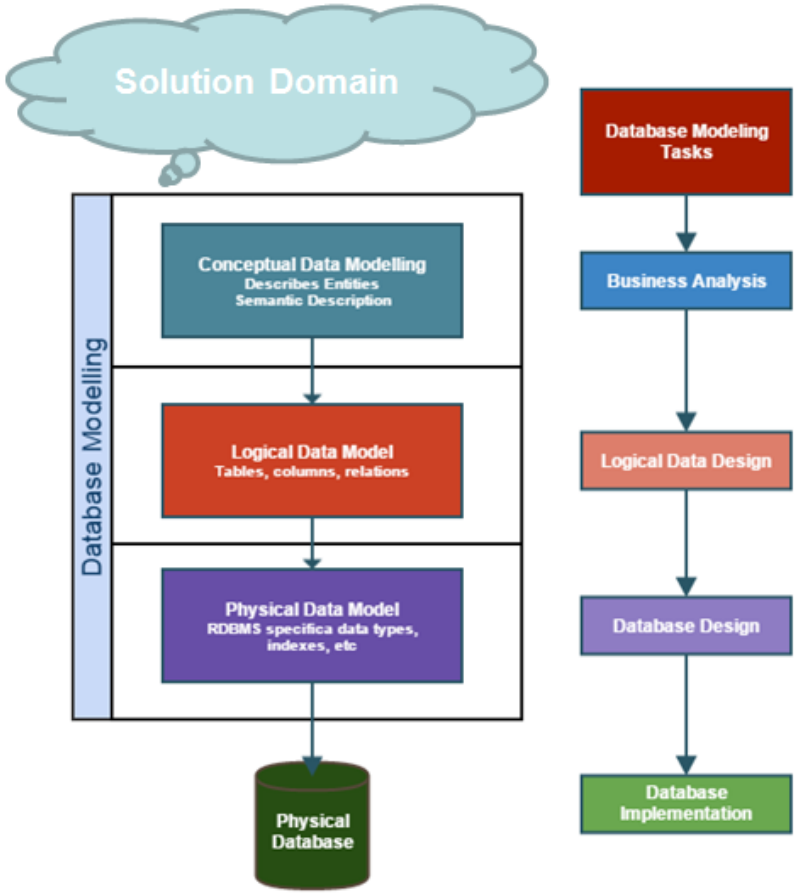
- Designing Databases
- Data Models
- Conceptual Design
- Logical Design
- Physical Design
- Implementation
- Example

Designing Database

- Many database problems are caused by poor database design
- Mission statement for the database project defines major aims of database application
  - Purpose of the database project
  - Path towards the efficient and effective creation of database system
- Once mission statement is defined, mission objectives are defined
- Each objective should identify a particular task that the database must support
- May be accompanied by some additional information that specifies:
  - The work to be done
  - The resources needed to do it
  - The money to pay for it all

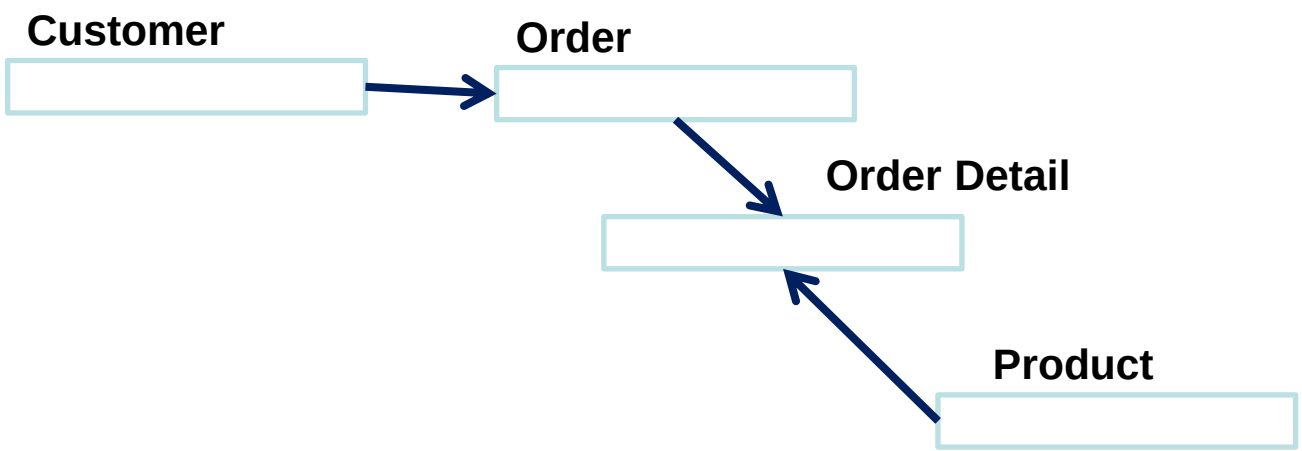
Data Models

Feature	Conceptual	Logical	Physical
Entity Names	✓	✓	
Entity Relationships	✓	✓	
Attributes		✓	
Primary Keys		✓	✓
Foreign Keys		✓	✓
Table Names			✓
Column Names			✓
Column Data Types			✓



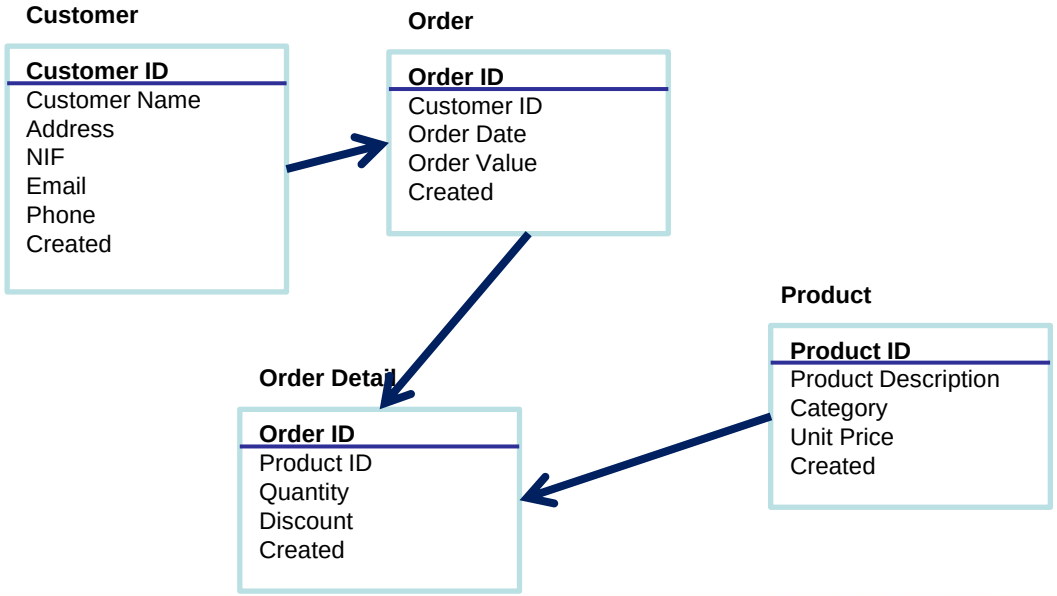
Conceptual Design

- Conceptual data model identifies the relationships between the different entities
- Features:
  - Includes the important entities and the relationships among them
  - No attribute is specified
  - No primary key is specified



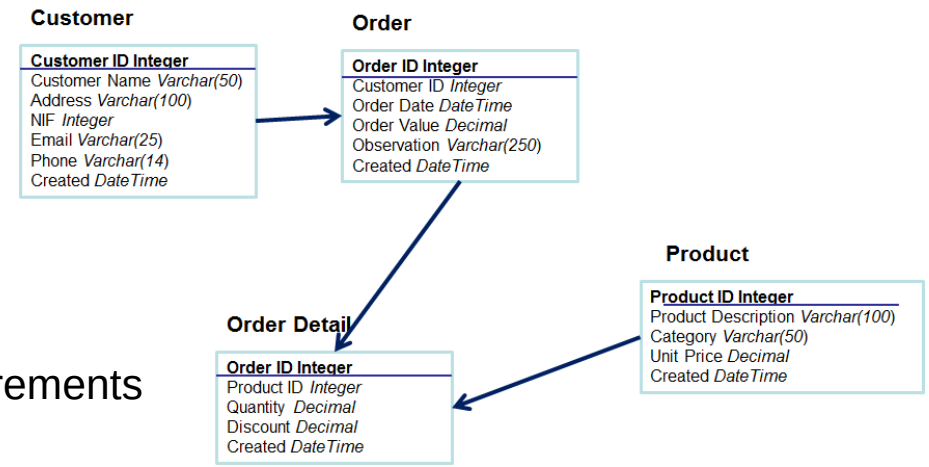
Logical Design

- Logical data model describes the detailed data
- Features:
  - Includes all entities and relationships among them
  - All attributes for each entity are specified
  - The primary key for each entity is specified
  - Foreign keys are specified
  - Uses normalization



Physical Design

- Represents how the model will be built in the database
- Features:
  - Specification of all tables and columns:
  - Foreign keys are used to identify relationships between tables.
  - Physical data model is specific to the RDBMS
- Steps:
  - Convert entities into tables
  - Convert relationships into foreign keys
  - Convert attributes into columns
  - Modify the physical data model based on physical constraints/requirements



Implementation

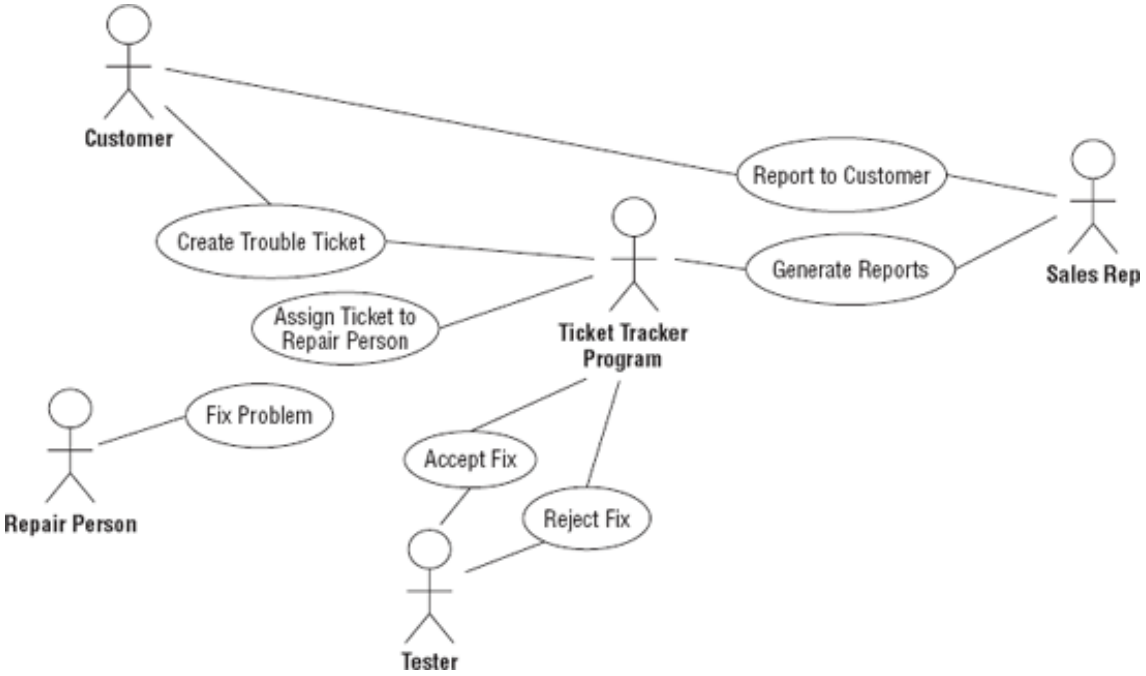
- Implementation
  - Is the physical realization of the database and application designs
  - Use data definition commands to create database, schemas and views
  - Use data manipulation language to load, manipulate and extract data
- Data Conversion and Loading
  - Transferring any existing data into new database
  - Update existing applications to run on new database.
  - Required when new database system is replacing an old system
  - Use DBMS utility or T-SQL
- Testing
  - Process of running the database system with intent of finding errors.
  - Demonstrates that database are working according to requirements.
  - Should also test usability of system

Database Design Technics

- Use Cases
- Entity-Relationship Model
- Database Modeling Tools

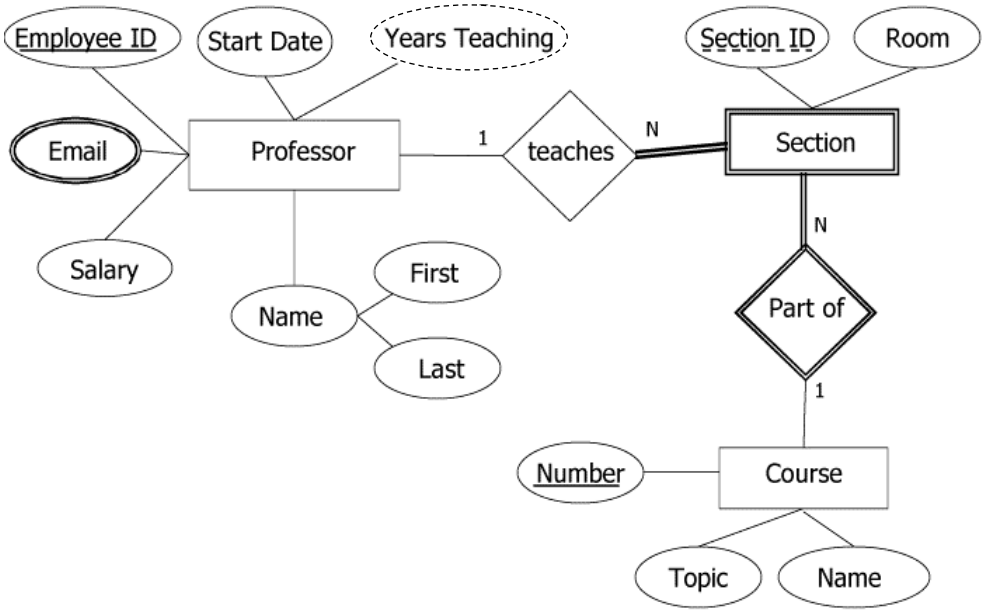
Use Cases

- Methodology used in system analysis to identify, clarify, and organize system requirements
- Is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal
- Elements:
  - Goals
  - Summary
  - Actors
  - Pre/post-conditions
  - Normal flow
  - Alternative flow
  - Notes



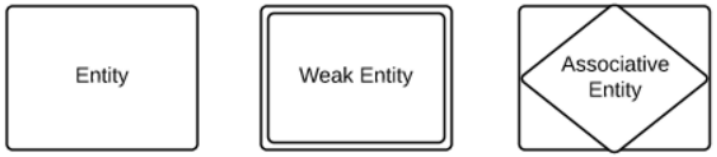
Entity-Relationship Model

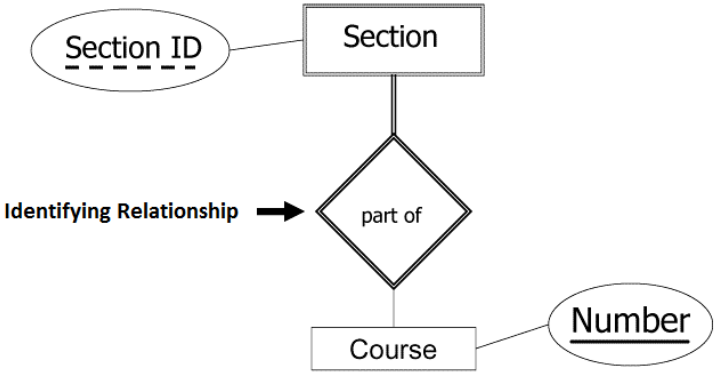
- Known as ER model, is the most popular conceptual model for database design
- It describes:
  - The data in a system and how that data is related
  - Data as entities, attributes and relationships



Entity-Relationship Model

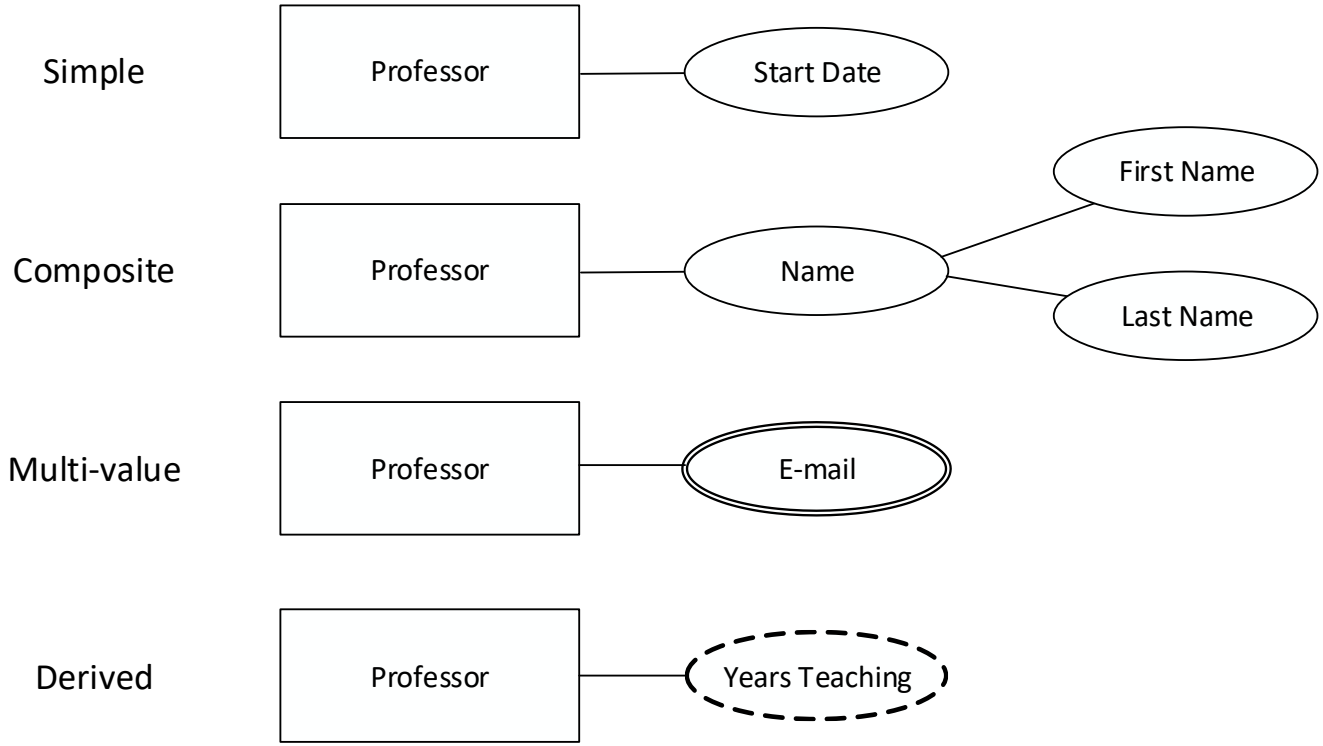
- Entities:
  - Are ER model basic object
  - Represents a “thing” with no independent existence
  - Can exist physically or conceptually: a teacher, a student, a course
- Types of entities:
  - Strong entities exist independently from other entity types
  - Weak entities depend on some other entity type and have no meaning in the diagram without depending on another entity
  - Associative entities are entities that associate the instances of one or more entity types





Entity-Relationship Model

- Attributes:
  - Each entity has a set of associated properties that describes it
  - Can be: simple, composite, derived, multi-value



Entity-Relationship Model

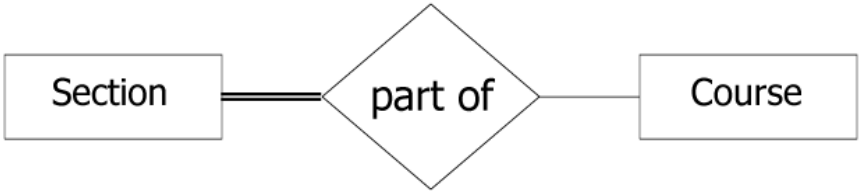
- Keys:
  - Candidate key: an attribute or set of attributes that uniquely identifies individual occurrences of an entity type
  - Primary key: an entity type may have one or more possible candidate keys, one of which is selected to be a primary key
  - Composite key: a candidate key that consists of two or more attributes.

Entity-Relationship Model

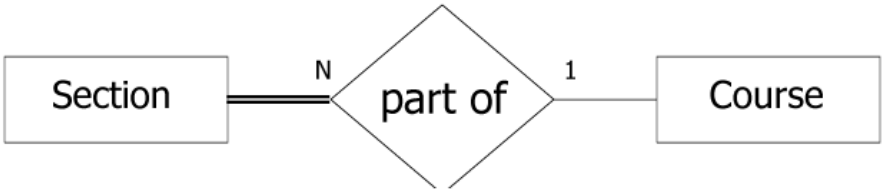
- Relationships:
  - Defines a set of associations between various entities
  - Can have attributes to define them
  - Are limited by:
    - Participation
    - Cardinality
- Degree of relationship: the number of participating entities in a relationship, e. g. binary, ternary etc.

Entity-Relationship Model

- Relationship participation:
  - Defines if the existence of an entity depends on it being related to another entity with a relationship type
    - Partial
    - Total



- Relationships cardinality:
  - The number of relationships that an entity may participate in
    - One-to-One: 1:1
    - One-to-Many: 1:M
    - Many-to-Many: M:M



Database Modeling Tools

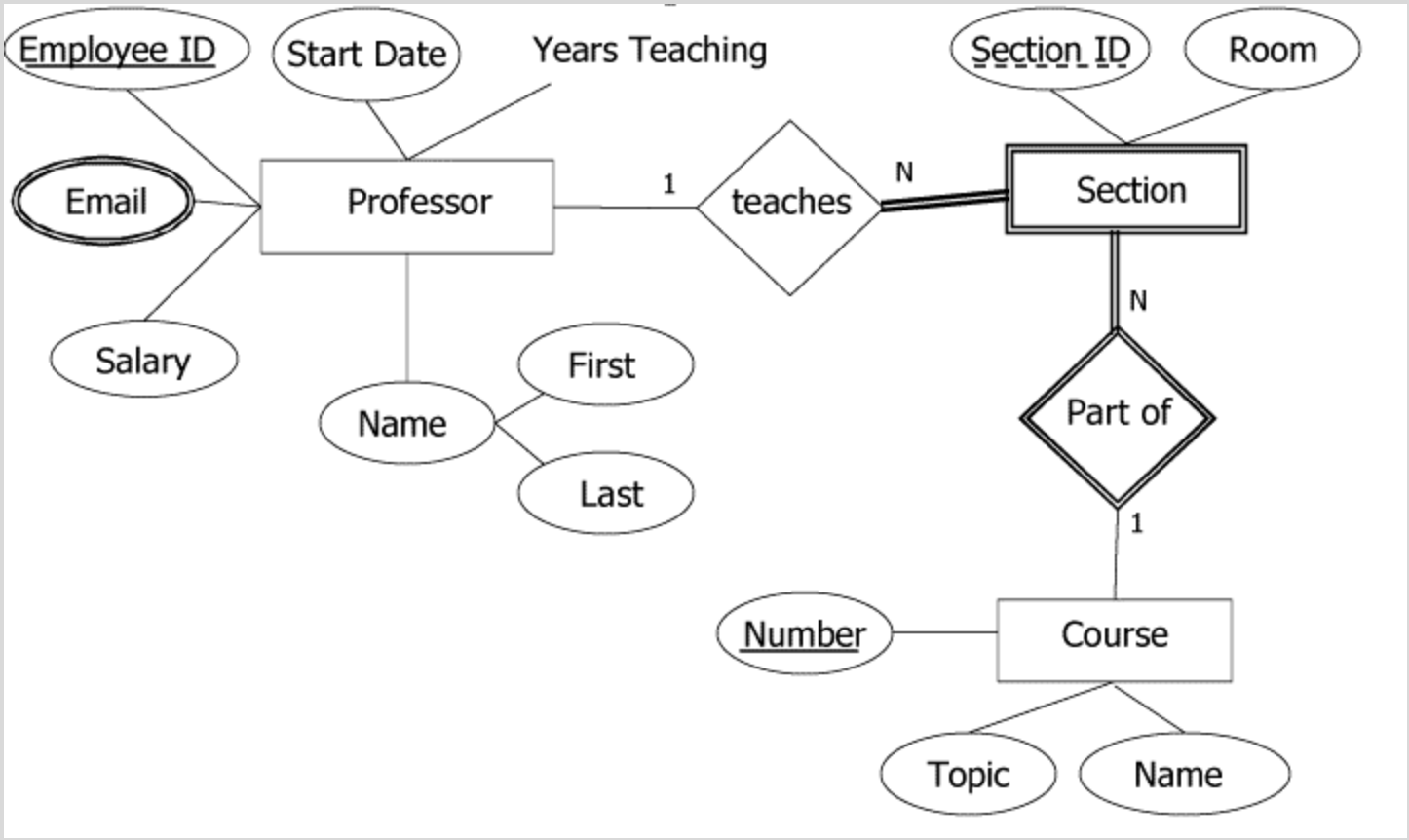
- Microsoft Visio
- UML Star
- Sparx System
- SmartDraw
- ConceptDraw
- …

Demo

- Theme:
  - Teaching Database
- Goal
  - Design an E-R schema for a database to store info about teachers, courses and course sections
- Specifications:
  - The name, employee ID number, salary and email address(es) of each teacher
  - How long each professor has been at the university
  - The course sections each professor teaches
  - The name, number and topic for each course offered
  - The section and room number for each course section
  - Each course section must have only one professor
  - Each course can have multiple sections

Demo

- Solution: E-R Diagram



Practice

- See exercise.

Summary

- Make sure you understand the problem.
- Write requirements documents to state the problem.
- Build use cases and/or ER to see if you have solved the problem.
- Design a solution.
- Test the design to see if it satisfies the use cases.
- Document everything.

Understanding Data Storage

Module 3

Overview

1. Normalization
2. Constraints
3. Indexes
4. Relationships
5. Practice
6. Summary

Normalization

- Normalization
- Normal Forms
- Demo
- Database Anomalies

Normalization

- Normalization:
  - Is a bottom-up technique for database design, normally based on an existing system
  - Is the process of organizing the columns and tables of a relational database to reduce data redundancy and improve data integrity
  - Is accomplished by applying normal forms
  - It will support data manipulation (insert, delete, update) without database anomalies
- Normal Forms (NFs):
  - There are five normalization forms (NFs)
  - Usually a relational database table is often described as "normalized" if it meets 3NF

Normal Forms

- 1NF: Eliminate repeating groups
  - Make a separate table for each set of related attributes
  - Give each table a primary key
- 2NF: Eliminate redundant data
  - If an attribute depends on only part of a multi-valued key remove it to a separate table
  - Give each table a primary key
- 3NF: Eliminate columns not dependent on key
  - All attributes must be directly dependent on the primary key
  - Eliminate transitive dependencies of non-key attributes upon other non-key attributes and not the primary key, moving them to another table
- 4NF: Isolate independent multiple relationships
  - Involves two independent attributes brought together to form a primary key along with a third attribute
- 5NF: Isolate semantically related multiple relationships
  - There may be practical constrains on information that justify separating logically related many-to-many relationships.

Demo:
Project Management Report

- The Project Management Report describes projects being worked upon by employees. This report is to be 'normalized‘.

Project Management Report

Project Code: PC010
Project Title: Pensions System

Project Manager: M Phillips **Project Budget:** £24,500

Employee No.	Employee Name	Department No.	Department Name	Hourly Rate
S10001	A Smith	L004	IT	£22.00
S10030	L Jones	L023	Pensions	£18.50
S21010	P Lewis	L004	IT	£21.00
S00232	R Smith	L003	Programming	£26.00

Total Staff on Project: 4 **Average Hourly Rate:** £21.88

Calculated Fields

Demo:
Project Management Report

- UNF: Unnormalized table
  - In order to obtain a UNF, select the data source (previous report) and convert into UNF
  - Steps:
    - Column headings for each data item on the report, ignoring calculated fields
    - Representative sample data into table
    - Identify a primary key for the table

Project Code	Project Title	Project Manager	Project Budget	Employee No.	Employee Name	Department No.	Department Name	Hourly Rate
PC010	Pensions System	M Phillips	24.500	S10001	A Smith	L004	IT	22,00
PC010	Pensions System	M Phillips	24.500	S10030	L Jones	L023	Pensions	18,50
PC010	Pensions System	M Phillips	24.500	S21010	P Lewis	L004	IT	21,00
PC045	Salaries System	H Martin	17.400	S10010	B Jones	L004	IT	21,75
PC045	Salaries System	H Martin	17.400	S10001	A Smith	L004	IT	18,00
PC045	Salaries System	H Martin	17.400	S31002	T Gilbert	L028	Database	25,50
PC045	Salaries System	H Martin	17.400	S13210	W Richards	L008	Salary	17,00
PC064	HR System	K Lewis	12.250	S31002	T Gilbert	L028	Database	23,25
PC064	HR System	K Lewis	12.250	S21010	P Lewis	L004	IT	17,50
PC064	HR System	K Lewis	12.250	S10034	B James	L009	HR	16,50

Demo:
Project Management Report

- 1NF: Eliminate repeating groups
  - Identify repeating attributes
  - Move those to a new table
  - Make a copy of the primary key into the new table

Project Code	Project Title	Project Manager	Project Budget
PC010	Pensions Syste	M Phillips	24.500
PC045	Salaries System	H Martin	17.400
PC064	HR System	K Lewis	12.250

Project Code	Employee No.	Employee Name	Department No.	Department Name	Hourly Rate
PC010	S10001	A Smith	L004	IT	22,00
PC010	S10030	L Jones	L023	Pensions	18,50
PC010	S21010	P Lewis	L004	IT	21,00
PC045	S10010	B Jones	L004	IT	21,75
PC045	S10001	A Smith	L004	IT	18,00
PC045	S31002	T Gilbert	L028	Database	25,50
PC045	S13210	W Richards	L008	Salary	17,00
PC064	S31002	T Gilbert	L028	Database	23,25
PC064	S21010	P Lewis	L004	IT	17,50
PC064	S10034	B James	L009	HR	16,50

Demo:
Project Management Report

- 2NF: Eliminate redundant data
  - Move all attributes that only depend on part of the table key to a new table
  - Make a copy of the non-key attribute upon which it is dependent and mark it as primary key

Project Code	Project Title	Project Manager	Project Budget
PC010	Pensions System	M Phillips	24.500
PC045	Salaries System	H Martin	17.400
PC064	HR System	K Lewis	12.250

Project Code	Employee No. (*)	Hourly Rate
PC010	S10001	22,00
PC010	S10030	18,50
PC010	S21010	21,00
PC045	S10010	21,75
PC045	S10001	18,00
PC045	S31002	25,50
PC045	S13210	17,00
PC064	S31002	23,25
PC064	S21010	17,50
PC064	S10034	16,50

Employee No.	Employee Name	Department No.	Department Name
S10001	A Smith	L004	IT
S10010	B Jones	L004	IT
S10030	L Jones	L023	Pensions
S10034	B James	L009	HR
S13210	W Richards	L008	Salary
S21010	P Lewis	L004	IT
S31002	T Gilbert	L028	Database

Demo:
Project Management Report

- 3NF: Eliminate columns not dependent on key
  - Create a new table with each non-key attribute that is more dependent on another non-key attribute than the table key itself
  - Make a copy of the non-key attribute upon which it is dependent and mark it as key
  - Leave the non-key attribute in the original table and mark it a foreign key (*)

Project Code	Project Title	Project Manager	Project Budget
PC010	Pensions System	M Phillips	24.500
PC045	Salaries System	H Martin	17.400
PC064	HR System	K Lewis	12.250

Project Code	Employee No. (*)	Hourly Rate
PC010	S10001	22,00
PC010	S10030	18,50
PC010	S21010	21,00
PC045	S10010	21,75
PC045	S10001	18,00
PC045	S31002	25,50
PC045	S13210	17,00
PC064	S31002	23,25
PC064	S21010	17,50
PC064	S10034	16,50

Employee No.	Employee Name	Department No. (*)
S10001	A Smith	L004
S10010	B Jones	L004
S10030	L Jones	L023
S10034	B James	L009
S13210	W Richards	L008
S21010	P Lewis	L004
S31002	T Gilbert	L028

Department No.	Department Name
L004	IT
L008	Salary
L009	HR
L023	Pensions
L028	Database

Database anomalies

- Database anomalies:
  - Are caused by redundancy and affect the process of inserting, deleting and modifying data
- Insertion anomaly:
  - Occurs when new record is inserted in a relation
  - A user cannot insert a fact about an entity until he has an additional fact about another entity
- Deletion anomaly:
  - Occurs when a record is deleted from the relation
  - The deletion of facts about an entity automatically deletes the fact of another entity
- Updation anomaly:
  - Occurs when the record is updated in the relation
  - The modification in the value of specific attribute requires modification in all records in which that value occurs

Constraints

- Data Integrity
- Constraints
- Key Constraints
- Oher Constraints

Data Integrity

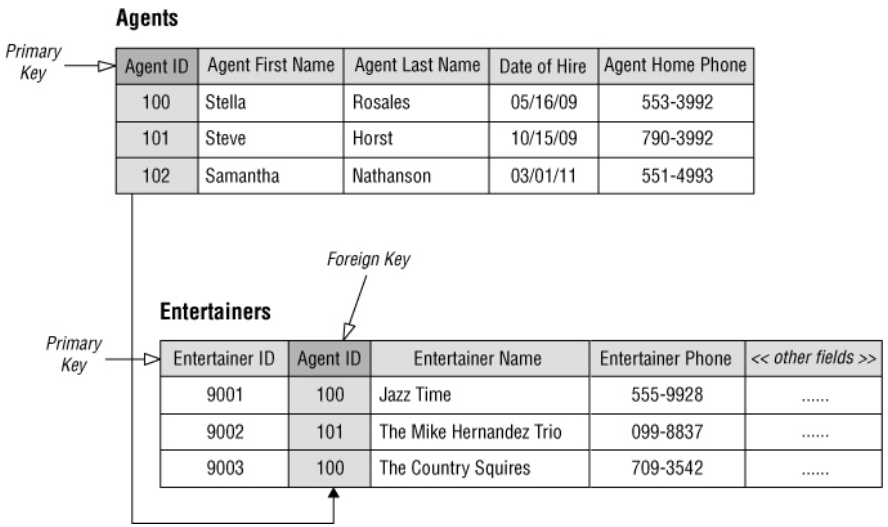
- Definition:
  - Data integrity ensures that the data within the database is reliable and adheres to business rules
- Categories:
  - Domain integrity: it ensures that the value meets a specified format and value criteria
  - Entity integrity: it ensures that every row in the table is uniquely identified
  - Referential integrity: it ensures that the data is consistent between related tables
  - User-defined integrity: it ensures that the values stored in the database remain consistent with established business policies

Constraints

- Database data integrity is achieved by constraints, which are field properties
- Keys are special fields with specific roles within a table and are also table constraints
- The other constraints assures data quality by reinforcing data integrity rules

Key Constraints

- Primary Key (PK):
  - Is a field or group of fields (composite PK) that uniquely identifies each record within a table
  - It enforces table-level integrity and helps establish relationships with other tables in the database
  - Every table has just one PK that must be unique and not null
- Foreign Key (FK):
  - Is a field that connects the primary table and the secondary table
  - The primary table PK is replicated to the secondary table and assumes the name of foreign key (FK)



Agent ID	Agent First Name	Agent Last Name	Date of Hire	Agent Home Phone
100	Stella	Rosales	05/16/09	553-3992
101	Steve	Horst	10/15/09	790-3992
102	Samantha	Nathanson	03/01/11	551-4993

Entertainer ID	Agent ID	Entertainer Name	Entertainer Phone	<< other fields >>
9001	100	Jazz Time	555-9928	.....
9002	101	The Mike Hernandez Trio	099-8837	.....
9003	100	The Country Squires	709-3542	.....

Oher Constraints

- Unique:
  - Accepts only unique values in column, as in PKs
- Not null:
  - Enforces a column to not accept null values, meaning that it must always contain a value, as in PKs
- Check:
  - Used to enforce domain integrity and to provide a means for restricting the values that can be entered in a column
- Default:
  - It’s not a real constraining mechanism, once it supplies a default value during an insertion if no value is given

Indexes

- Indexes
- Clustered
- Non-Clustered
- Columnstore

Indexes

- Indexes are physical structures used to optimize data processing
- Indexes allow you to access data faster, but there is also a cost associated with indexes because they slow down insert, update, and delete operations
- Major SQL indexes:
  - Clustered
  - Non-Clustered
  - Columnstore

Clustered

- Each table can have only on clustered index that defines how SQL Server will sort the data stored inside the table
- Clustered index is a physical construct, as opposed to most indexes which are logical

Non-Clustered

- You have the freedom to create your own non-clustered indexes; they have a structure separate from what is found in the clustered index structure.
- A non-cluster index does not contain the entire data row, it contains just the columns defined in the index and a pointer or key to the actual data row
- Single table supports up to 999 non-cluster indexes

Columnstore

- SQL Server 2012 → nonclustered columnstore indexes
- SQL Server 2014 → both clustered and nonclustered columnstore indexes
- An in-memory columnstore index stores and manages data by using column-based data storage and column-based query processing
- Columnstore indexes work well for data warehousing workloads that primarily perform bulk loads and read-only queries
- Use the columnstore index to achieve up to 10x query performance gains over traditional row-oriented storage, and up to 7x data compression over the uncompressed data size

Relationships

- Structure-related Terms
- Relationship-related Terms
- One-to-Many Relationship
- Many-to-Many Relationship
- One-to-One Relationship

Structure-related Terms

- Table
  - Relation of data
  - Each relation is composed of tuples (records) and attributes (fields)
- Field
  - Known as attribute in relational database theory
  - The smallest structure in the database
  - The structure that actually stores data
- Record
  - Represents a unique instance of the subject of a table
  - It is composed of the entire set of fields in a table, regardless of whether the fields contain values
- Keys
  - Keys are fields that play specific roles within a table
  - The most significant keys are the primary and foreign key

Relationship-related Terms

- Relationships:
  - It is crucial to data integrity because it helps reduce redundant data and eliminate duplicate data
  - A relationship works by matching data in key columns, usually with the same name in both tables
  - The relationship matches the primary key (PK) from one table with an entry in the foreign key (FK) in the other table
  - For example, book sales can be associated with the specific titles sold by creating a relationship between the Title.TitleID (PK) and Sale.TitleID (FK)
- Types of relationships:
  - One-to-one
  - One-to-many
  - Many-to-many

One-to-Many Relationship

- Definition:
  - Is the most common type of relationship in relational databases
  - A row in table A can have many matching rows in table B, but a row in table B can have only one matching row in table A
  - Example: the publishers and titles tables have a 1-n relationship: each publisher produces many titles, but each title if from only one publisher
- Process:
  - Make a 1-n relationship if only one of the related columns is a PK or has a unique constraint

Agents

Agent ID	Agent First Name	Agent Last Name	Date of Hire	Agent Home Phone
100	Stella	Rosales	05/16/09	553-3992
101	Steve	Horst	10/15/09	790-3992
102	Samantha	Nathanson	03/01/11	551-4993

Entertainers

Entertainer ID	Agent ID	Entertainer Name	Entertainer Phone	*<< other fields >>*
9001	101	Jazz Time	555-9928	.....
9002	100	The Mike Hernandez Trio	959-8837	.....
9003	100	The Country Squires	709-3542	.....

Many-to-Many Relationship

- Definition:
  - A row in table A can have many matching rows in table B, and vice versa
  - Example: the authors table and the titles table have a n-n relationship that is defined by a 1-n relationship from each of these tables to the new table, TitleAuthor
- Process:
  - You create such a relationship by defining a third table, called a junction table, whose PK consists of the FK from both table A and table B
  - The PK of the TitleAuthor table is the combination of the AuthorID column (PK) and the TitleID column (PK)

Many-to-Many Relationship

Students

Student ID	Student First Name	Student Last Name	Student Phone	*<< other fields >>*
60001	Zachary	Erlich	553-3992	.....
60002	Susan	Black	790-3992	.....
60003	Joe	Rosales	551-4993	.....

Classes

Class ID	Class Name	Instructor ID	*<< other fields >>*
900001	Intro. to Political Science	220087	.....
900002	Adv. Music Theory	220039	.....
900003	American History	220148	.....

Students

Student ID	Student First Name	Student Last Name	Student Phone	*<< other fields >>*
60001	Zachary	Erlich	553-3992	.....
60002	Susan	Black	790-3992	.....
60003	Joe	Rosales	551-4993	.....

Student Schedule (*Linking Table*)

Student ID	Class ID
60003	900001
60001	900003
60003	900003
60002	900002
60001	900001

Classes

Class ID	Class Name	Instructor ID	*<< other fields >>*
900001	Intro. to Political Science	220087	.....
900002	Adv. Music Theory	220039	.....
900003	American History	220148	.....

One-to-One Relationship

- Definition:
  - A row in table A can have no more than one matching row in table B, and vice versa
  - A 1-1 relationship is created if both of the related columns are PKs
  - 1-1 relationship is not common because most information related in this way would be all in one table
- Process:
  - Link both tables by their PK

Employees

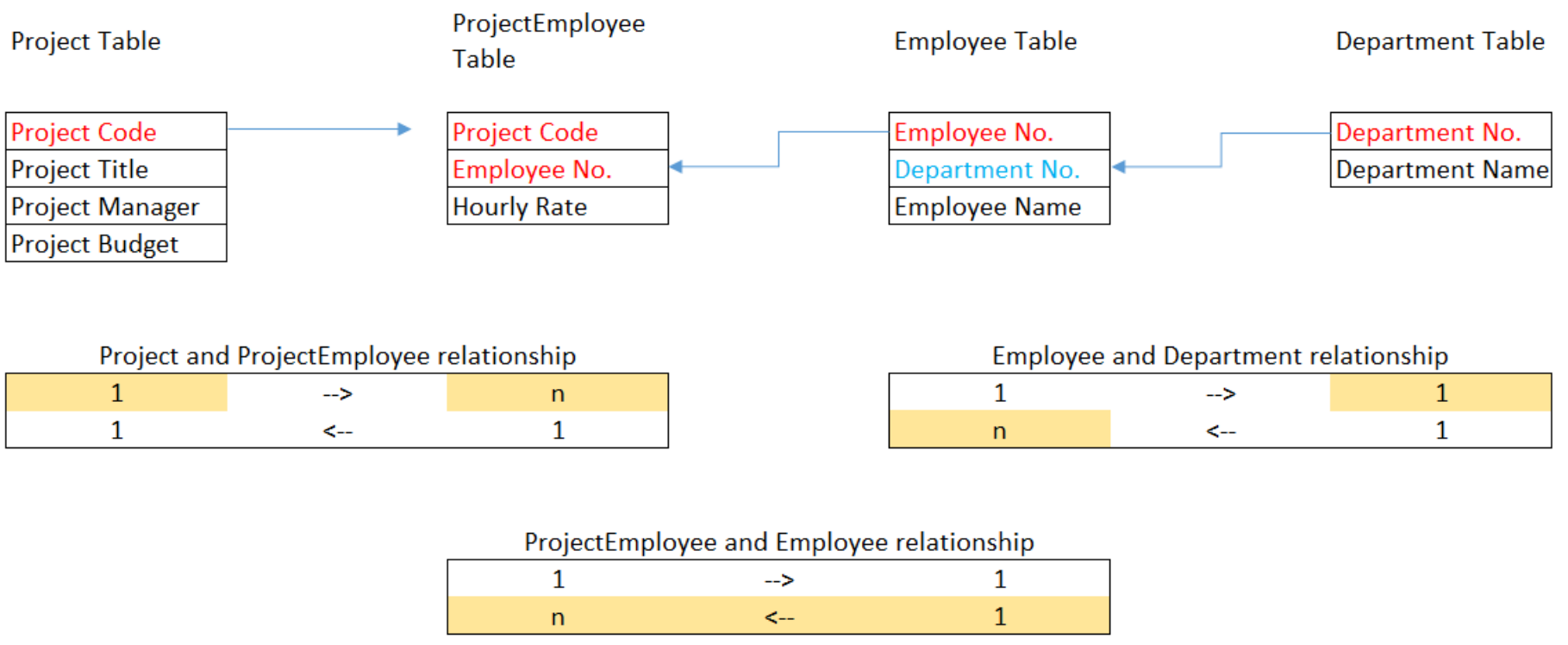
Employee ID	Employee First Name	Employee Last Name	Home Phone	*<< other fields >>*
100	Zachary	Erlich	553-3992	.....
101	Susan	Black	790-3992	.....
102	Joe	Rosales	551-4993	.....

Compensation

Employee ID	Hourly Rate	Commission Rate	*<< other fields >>*
100	19.75	3.5%	.....
101	25.00	5.0%	.....
102	22.50	5.0%	.....

Demo:
Project Management Report

- Relationships:



Practice

- See exercise.

Summary

- In order to work with relational databases, data must be normalized
- Normalization is a bottom-up technique for database design that applies formal rules to data
- In a database system we use constraints to specify rules for the data in a table
- We use indexes to improve the speed of data retrieval operations on a database table
- When implementing the database physical model we must relate the tables using well structured relationships

Introducing Microsoft SQL Server

Module 4

Overview

1. Microsoft SQL Server
2. T-SQL
3. Demo
4. Practice
5. Summary

Microsoft SQL Server

- What is SQL Server?
- Components
- Features
- Database Engine
- Database Transaction
- SQL Server Management Studio
- Database Files
- SQL Server 2016

What is SQL Server?

- RDBMS:
  - Is an enterprise-class database management system
  - Is Microsoft Relational Database Management System (RDBMS)
  - Is a database server capable of running anything from a personal database to a multiserver database system, managing terabytes of information
- It works with relational tables, which properties are:
  - The values are atomic
  - Each row is unique
  - Column values are of the same kind
  - Each column must have a unique name
  - The sequence of columns is insignificant
  - The sequence of rows is insignificant

Components

- SQL Server Database Engine
  - SQL Server Management Studio
- SQL Server Integration Services → SSIS
  - Master Data Services
  - Data Quality Services
- SQL Server Analysis Services → SSAS
  - OLAP cubes
  - Data mining
- SQL Server Reporting Services → SSRS
  - PowerPivot

Features

- **RDBMS**
  - Manageability
  - Management Tools
  - Security
  - Replication
  - High Availability
  - Scalability and Performance
- **Integration Services (SSIS)**
  - Data Quality Services
  - Master Data Services
- **Reporting Services (SSRS)**
  - Power Pivot for SharePoint
- **Analysis Services (SSAS)**
  - Data Warehouse
  - Data Mining
  - BI Semantic Model
    - Multi Dimensional
    - Tabular
- **Development**
  - Development Tools
  - Programmability
- **Special**
  - Spatial and Location Services
  - Database mail
  - StreamInsight

See in MSDN

Database Engine

- The SQL Server Database Engine is a Windows service that can be used to store and process data in a relational format
- SQL Server Database Engine offers:
  - Reliable storage for data
  - Means to rapidly retrieve data
  - Consistent access to data
  - Control access to data through security
  - Enforces data integrity to ensure that the data is reliable and consistent
- Supported databases:
  - OLTP (On-line Transaction Processing) supports database transactions
  - OLAP (On-line Analytical Processing) deals with data analysis

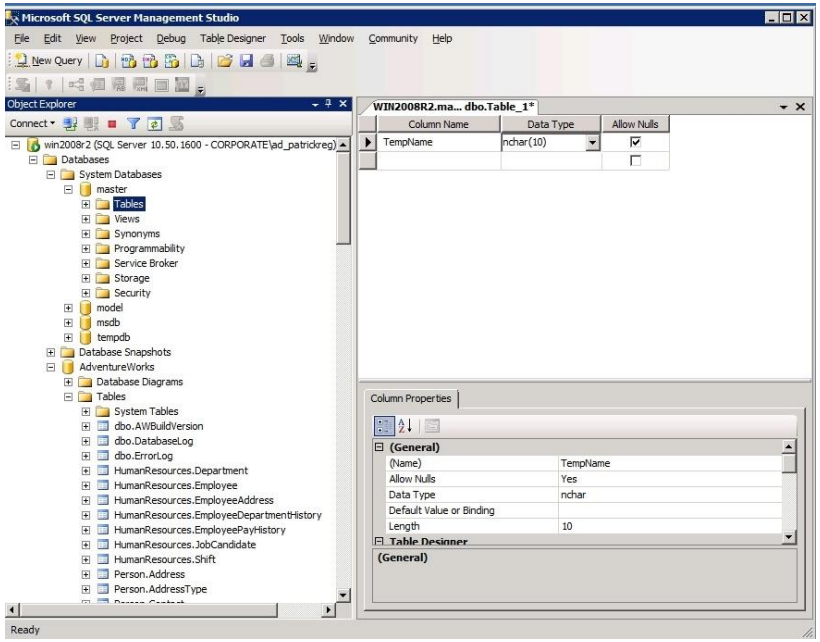
Database Transaction

- Database transaction
  - Is a sequence of database operations that satisfies the ACID properties
  - For example, a transfer of funds from one bank account to another is a single transaction
- ACID
  - Is an acronym for Atomicity, Consistency, Isolation, Durability
  - Is a set of transaction properties that guarantees database reliability

Atomicity	Requires that each transaction be all or nothing
Consistency	Guarantees that a transaction never leaves your database in a half-finished state
Isolation	Keeps transactions separated from each other until they are finished
Durability	Ensures that once a transaction has been committed, it will remain so

SQL Server Management Studio

- In short, SSMS
- SQL Server Management Studio is used to manage database engine
- The central feature of SSMS is the Object Explorer, which allows the user to browse, select and manage any of the objects within the server



Database Files

- SQL server uses 3 types of files to store databases:
  - Primary data files (.mdf) which contain user-defined objects, such as tables and views, as well as system tables
  - Secondary data files (.ndf) on separate physical disks
  - Transaction log files (.ldf) don't contain objects such as tables or views, but the actual transaction record
- Primary data file:
  - Database files are broken up into 8KB pages
  - Each data page stores rows until they reach 8KB size
- Secondary data file:
  - Used to split data among disks
- Log file:
  - It's optional and depends on database's Recovery Model
  - Very useful in disaster recovery situations

SQL Server 2016

- Versions:
  - SQL Server 7.0, 2000, 2005, 2008, 2008 R2, 2012, 2014, 2016
- Editions:
  - Enterprise
  - Standard
  - Express
  - Developer
- Versions: http://sqlserverbuilds.blogspot.pt/

T-SQL

- T-SQL
- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)
- Transaction Control Language (TCL)
- T-SQL Summary
- SQL Query Editor
- sqlcmd Utility

T-SQL

- T-SQL (Transact-SQL) is an extension of SQL language
- T-SQL is central to using SQL Server, since all applications that communicate with an instance of SQL Server do so by sending T-SQL statements to the server, regardless of the user interface of the application
- T-SQL scripts are written in a text editor, preferably in SQL Server Query Editor
- These T-SQL scripts can be saved in .sql files
- The scripts are structured in statements organized by categories
- T-SQL language is divided into 4 subsets:
  - Data Manipulation Language (DML)
  - Data Definition Language (DDL)
  - Data Control Language (DCL)
  - Transaction Control Language (TCL)

Data Definition Language (DDL)

- DDL deals with creating and manipulating database objects like tables, constraints, views and stored procedures
- Core statements:

CREATE	Creates a SQL Server database object (table, view or stored procedure)
ALTER	Changes an existing object
DROP	Removes an object from the database

```sql
-- CREATE
CREATE TABLE Person(
    PersonID INT IDENTITY (1,1) CONSTRAINT PK_PersonID PRIMARY KEY,
    FirstName NVARCHAR(20),
    LastName NVARCHAR(25)
);

-- ALTER
ALTER TABLE Person
ADD BirthDate DATETIME

-- DROP
DROP TABLE Person
```

Data Manipulation Language (DML)

- DML is used for manipulating data
- Core statements:

SELECT	Select/read records from database tables
INSERT	Insert new records in table
UPDATE	Update existing records
DELETE	Delete existing records

```sql
-- SELECT
SELECT *
FROM employee

-- INSERT
INSERT INTO employee(emp_id, fname, minit, lname,
                     job_id, job_lvl, pub_id, hire_date)
            VALUES('0000000', 'Almir', 'M', 'Vuk',
                     7, 12, 1207, 2009-05-09)

-- UPDATE
UPDATE employee
SET fname = 'ALMIR'
WHERE emp_id = '0000000'

-- DELETE
DELETE
FROM employee
WHERE emp_id = '0000000'
```

Data Control Language (DCL)

- DCL is used for securing data (authorization and permissions)
- Core statements:

GRANT	To give permissions on database objetcs to users
DENY	To deny permissions to users
REVOKE	To take back permission from users

```sql

-- GRANT
GRANT SELECT, INSERT, UPDATE, DELETE ON Employees TO almir

-- REVOKE
REVOKE INSERT ON Employees TO almir

-- DENY
DENY UPDATE ON Employees TO almir

```

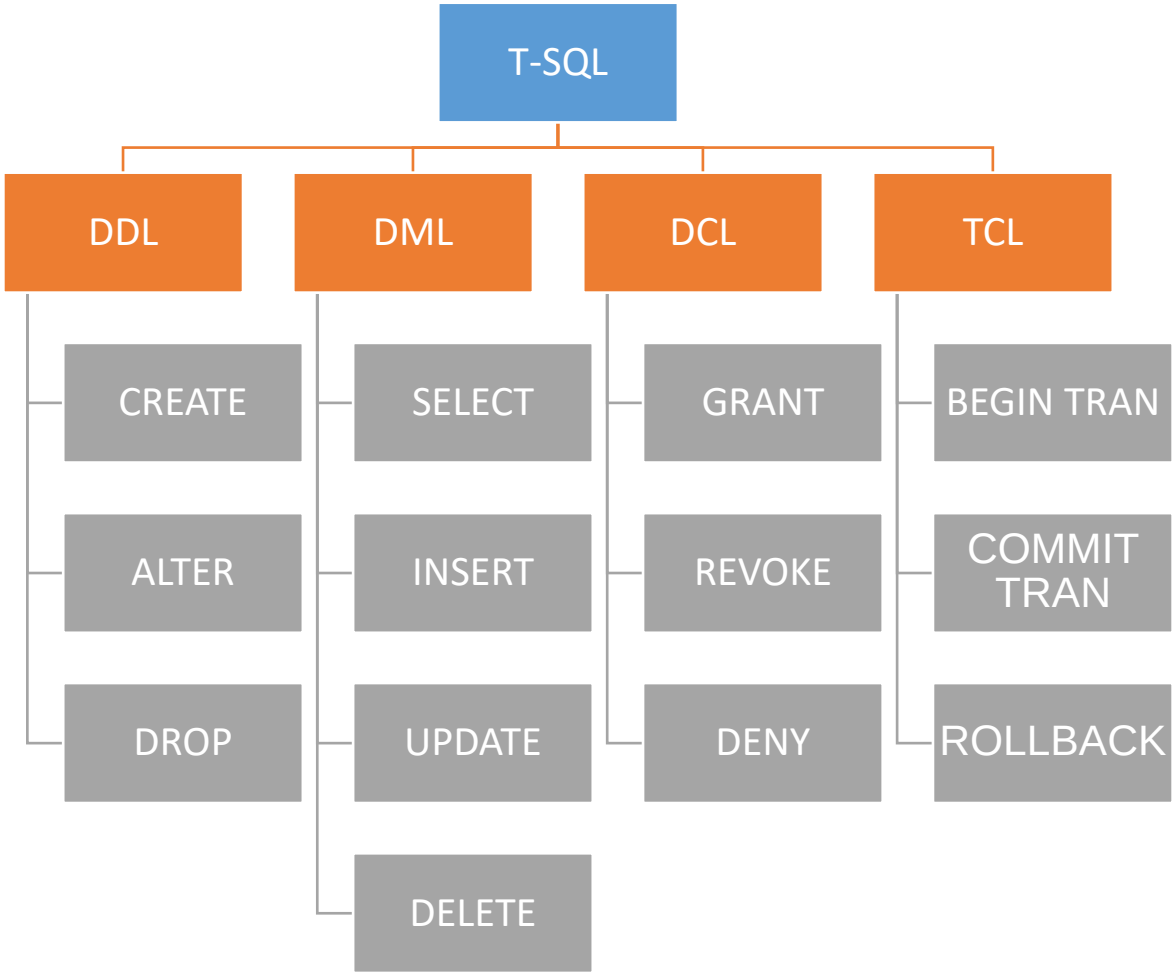
Transaction Control Language (TCL)

- TCL is used for manipulating transactions
- Core statements:

BEGIN TRAN	Begin of transaction
COMMIT TRAN	Commit for completed transaction
ROLLBACK	Go back to beginning if something went wrong in transaction

```sql
BEGIN TRANSACTION
UPDATE dbo.authors
SET au_fname = 'Almir'
WHERE au_id = '172-32-1176'

UPDATE authors
SET au_fname = 'Almir'
WHERE city = 'Mostar'

IF @@ROWCOUNT = 5
    COMMIT TRANSACTION
ELSE
    ROLLBACK TRANSACTION
```

T-SQL Summary



SQL Query Editor

- Use the Database Engine Query Editor to create and run scripts containing T-SQL statements

SQLQuery1.sql - MRSPC.Northwind (MRSPC\MRS (60))*

```
USE [Northwind];
GO

SELECT TOP(4) [ContactName], [City], [Phone] FROM [dbo].[Customers]
GO
```

100 %

Results | Messages

	ContactName	City	Phone
1	Maria Anders	Berlin	030-0074321
2	Ana Trujillo	México D.F.	(5) 555-4729
3	Antonio Moreno	México D.F.	(5) 555-3932
4	Thomas Hardy	London	(171) 555-7788

Query executed successfully. | MRSPC (12.0 SP1) | MRSPC\MRS (60) | Northwind | 00:00:00 | 4 rows

sqlcmd Utility

- sqlcmd utility is a Microsoft Win32 command prompt utility
- The sqlcmd utility lets you run ad-hoc T-SQL statements and scripts

SQLCMD

```
1> USE [Northwind];
2> GO
Changed database context to 'Northwind'.
1> SELECT TOP(4) [ContactName], [City], [Phone] FROM [dbo].[Customers]
2> GO
ContactName                    City           Phone
------------------------------ -------------- ------------------------
Maria Anders                   Berlin         030-0074321
Ana Trujillo                   México D.F.    (5) 555-4729
Antonio Moreno                 México D.F.    (5) 555-3932
Thomas Hardy                   London         (171) 555-7788

(4 rows affected)
1>
```

Demo

- Open SQL Server
- Explore SSMS
- Explore Projects and Solutions
- Explore AdventureWorks database
  - Objects
  - Files
  - Options
- Attach and Detach Northwind database
- Explore the query editor
- Use the sqlcmd utility

Practice

- See exercise.

Summary

- SQL Server is one of the most complete and robust relational database management system
- Analyze carefully SQL Server features and componentes to choose the right version and edition for your corporation
- Each database is stored in a disk and it’s data is spreaded across files
- T-SQL is the language used to manipulate SQL Server objects and data and is divided into 4 subsets: DDL, DML, DCL and TCL
- Database Engine is the core of SQL Server
- SSMS is the environment used to administer the server, manipulate databases and work with queries

Creating Database Objects

Module 5

Overview

1. SQL Server Objects
2. Databases
3. Tables
4. Demo
5. Practice
6. Summary

SQL Server Objects

- System Databases
- System Tables
- Views
- Stored Procedures

System Databases

- master:
  - Contains database objects: system tables, stored procedures, etc.
  - Includes security and configuration information for all databases
  - Is critical to the correct operation of the entire SQL Server instance
- msdb:
  - Is used by SQL Server Agent for scheduling alerts and jobs
- model:
  - Is a template for all databases that you create
- resource:
  - Read-only and hidden db that contains SQL Server system objects
  - These objects logically appear in the sys schema
- tempdb:
  - It holdes temporary objects or intermediate result sets
  - Each time the SQL Server service is started, tempdb is rebuilt

System Tables

- System views belong to the sys schema
- Some of these system tables include:
  - sys.Tables
  - sys.Columns
  - sys.Databases
  - sys.Constraints
  - sys.Views
  - sys.Procedures
  - sys.Indexes
  - sys.Triggers
  - sys.Objects

Views

- A view is simply a virtual table consisting of different columns from one or more tables
- Unlike a table, a view is stored in the database as a query object; therefore, a view is an object that obtains its data from one or more tables
- It´s an abstraction layer for tables relationship complexity
- Database > Views

```sql
USE AdventureWorks2016;
GO

SELECT * FROM HumanResources.vEmployee;
GO
```

Stored Procedures

- A stored procedure is a an written T-SQL statement which has been saved into the database
- One of the things that will save you time when running the same query over and over again is to create a stored procedure, which you can then execute from within the database's command environment
- Database > Programmability > Stored Procedures

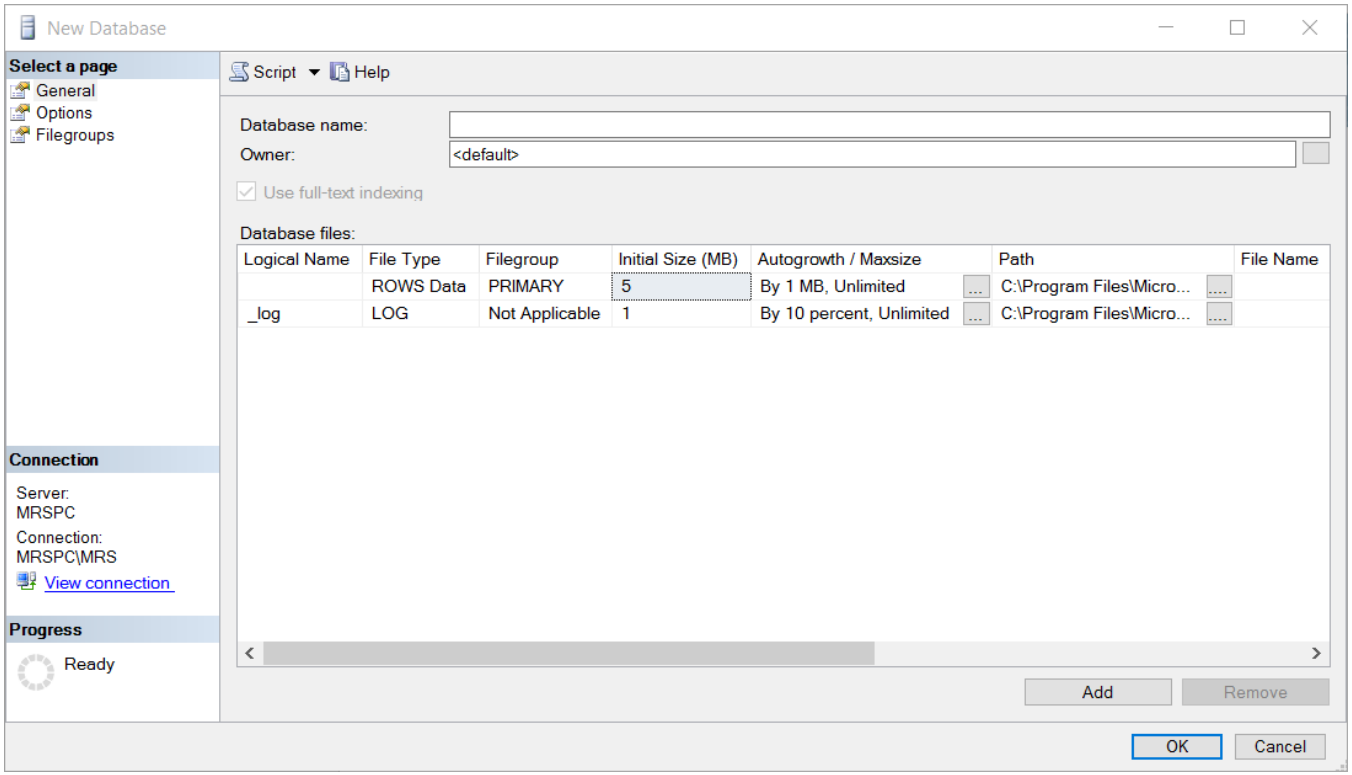
```sql
USE AdventureWorks2016;
GO

EXEC uspGetOrderTrackingByTrackingNumber 'EE33-45E8-9F';
GO
```

Databases

- Creating Databases using SSMS
- Creating Databases using T-SQL
- Database Schemas

Creating Databases using SSMS

- SSMS helps the creation of databases
- Steps: SSMS > Object Explorer > Databases right click > New Database



Creating Databases using SSMS

- General page:
  - Database name: will be used as the name for the data and log files
  - Owner: the database owner (dbo) has full administration rights on the database
  - Database files:

Logical Name	It's a friendly name for the database
File Type	The type of files that will be generated: Rows Data for the data file (MDF) and Log for the log file (LDF)
Filegroup	It groups database objects and files together for allocation and administration purposes
Path	The physical files location
Initial Size (MB)	Database's initial size in MB
Autogrowth / Maxsize	Database's options for autogrowth and maxsize
Path	Physical path for saving database's files
File Name	The physical name of the database files

Creating Databases using T-SQL

- CREATE DATABASE statement
- Syntax and example :

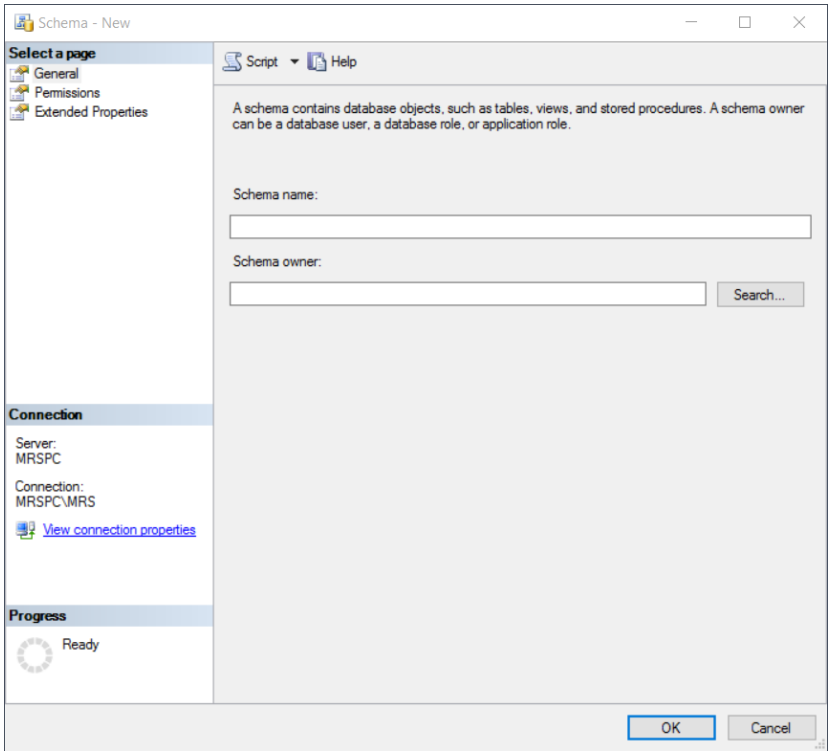
```
CREATE DATABASE database_name
[ CONTAINMENT = { NONE | PARTIAL } ]
[ ON
      [ PRIMARY ] <filespec> [ ,...n ]
      [ , <filegroup> [ ,...n ] ]
      [ LOG ON <filespec> [ ,...n ] ]
]
[ COLLATE collation_name ]
[ WITH  <option> [,...n ] ]
[;]
```

```sql
CREATE DATABASE TesteDB
ON  PRIMARY
(
  NAME = N'TesteDB',
  FILENAME = N'C:\Data\TesteDB.mdf',
  SIZE = 5120KB,
  MAXSIZE = UNLIMITED,
  FILEGROWTH = 1024KB
)
LOG ON
(
  NAME = N'TesteDB_log',
  FILENAME = N'C:\Log\TesteDB_log.ldf',
  SIZE = 1024KB,
  MAXSIZE = 2048GB,
  FILEGROWTH = 10%
);
GO
```

Database Schemas

- Schema is a database object that allows you to logically group objects, manage and define ownership, independently of the individual users in the database
- The schemas are located in: Database > Security > Schemas
- Example:

```
USE TesteDB;
GO
CREATE SCHEMA controlo AUTHORIZATION ABDBI;
GO
```



Tables

- Data Types
- Data Conversion
- Creating Tables Using SSMS
- Creating Tables Using T-SQL
- Manipulating Tables

Data Types

- In SQL Server, each column, local variable, expression, and parameter has a related data type
- A data type is an attribute that specifies the type of data that the object can hold
- Data types categories:
  - Exact numerics
  - Approximate numerics
  - Date and time
  - Character strings
  - Unicode character strings
  - Binary strings
  - Other data types

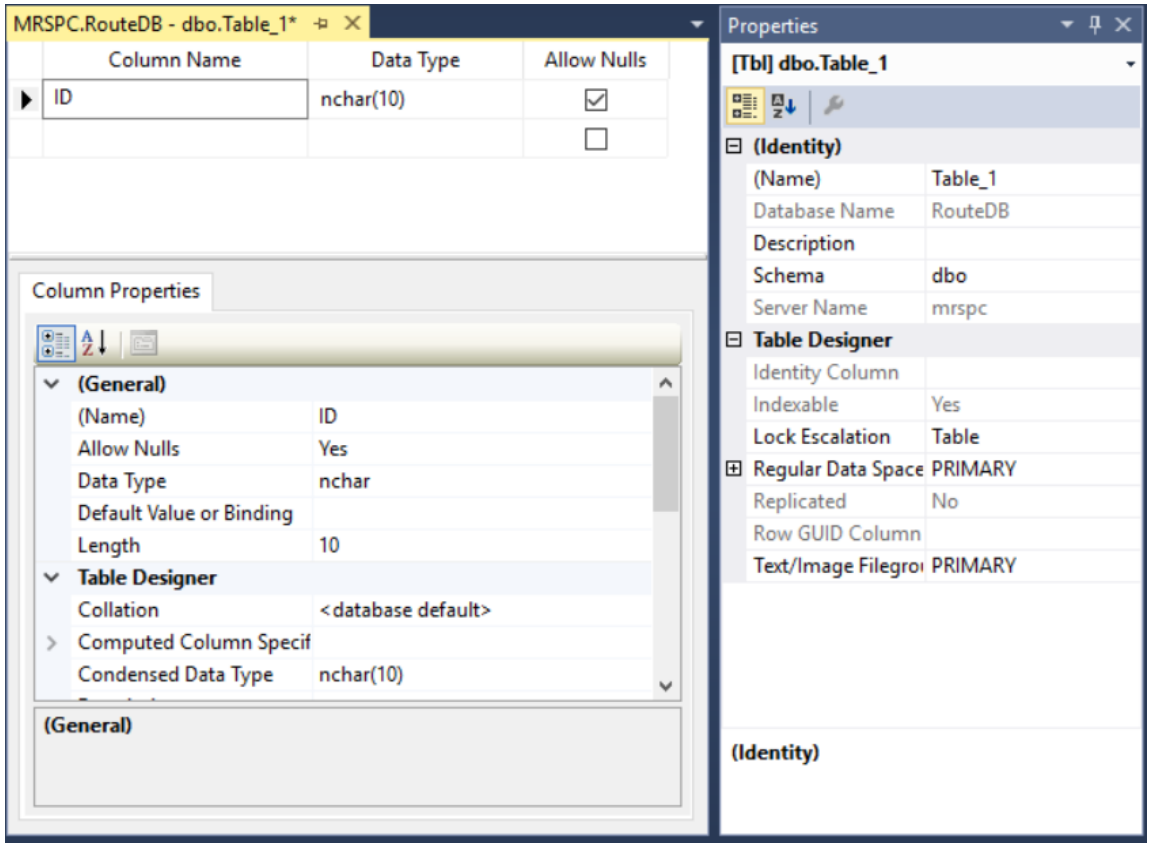
See resume table

Data Conversion

- Need for data types conversion:
  - When data from one object is moved to, compared with, or combined with data from another object, the data may have to be converted from the data type of one object to the data type of the other
  - When data from a Transact-SQL result column, return code, or output parameter is moved into a program variable, the data must be converted from the SQL Server system data type to the data type of the variable
- Types of conversions: implicitly or explicitly
  - Implicit conversions are not visible to the user, since SQL Server automatically converts the data from one data type to another
  - Explicit conversions use the CAST or CONVERT functions

Creating Tables Using SSMS

- Steps: SSMS > Object Explorer > Database > Tables > Right-click > New Table > Define table columns



Creating Tables Using T-SQL

- CREATE TABLE statement

```
      --Simple CREATE TABLE Syntax (common if not using options)
CREATE TABLE
    [ database_name . [ schema_name ] . | schema_name . ] table_name
    ( { <column_definition> } [ ,...n ] )
[ ; ]
```

```
USE TesteDB;
GO

CREATE TABLE dbo.Person
(
     ID int IDENTITY (1,1) NOT NULL,
     [Name] nvarchar(70) NOT NULL,
     BirthDate date,
     CONSTRAINT PK_Person_ID PRIMARY KEY CLUSTERED (ID ASC)
);
GO
```

Manipulating Tables

- Alter:
  - Using SSMS: Object Explorer > Database > Table > Right-click > Modify > Design > …
  - Using T-SQL:

```
ALTER TABLE dbo.Person
  ADD CONSTRAINT PK_Person_ID
  PRIMARY KEY CLUSTERED(ID ASC);
GO
```

- Drop:
  - Using SSMS: Object Explorer > Database > Table > Right-click > Delete
  - Using T-SQL:

```
DROP TABLE dbo.Person;
GO
```

Demo

- Using SSMS, create scripts for:
  - Create TestDB database
  - Create a new schema named HR
  - In HR schema, create the table Person with: PersonID (int, identity, PK), Name (nvarchar(70), not null), BirthDate (datetime)
  - Change previous table: delete BirthDate and create PhoneNumber (varchar(9) and Notes (nvarchar(MAX))
  - Drop Person table
  - Drop database TestDB

Practice

- See exercise.

Summary

- In order to create and manipulate database objects, we use SSMS or script it with T-SQL
- Using right-click it’s possible to do almost all operations with database objects
- Microsoft exams usualy refer to T-SQL statements

Manipulating Data

Module 6

Overview

1. Select Data
2. Select Data with Multiple Tables
3. Insert, Update and Delete Data
4. Demo
5. Practice
6. Summary

Select Data

- SELECT Statement
- Syntax
- T-SQL Operators
- SELECT…FROM
- SELECT…INTO
- SELECT…WHERE
- SELECT…GROUP BY
- SELECT…HAVING
- SELECT…ORDER BY

SELECT Statement

- The SELECT statement is used extract data from tables or views
- A simple SELECT uses one table but is possible to combine several tables using JOINs and INTERSECTs
- To build a correct SELECT you must provide:
  - Tables to retrieve data from
  - Columns to show
  - Conditions that data must comply (optional)
  - Output order (optional)

Syntax

```
SELECT select_list [ INTO new_table ]
[ FROM table_source ] [ WHERE search_condition ]
[ GROUP BY group_by_expression ]
[ HAVING search_condition ]
[ ORDER BY order_expression [ ASC | DESC ] ]
```

- The clauses are logically processed in the following order:
  1. FROM
  2. WHERE
  3. GROUP BY
  4. HAVING
  5. SELECT
  6. ORDER BY

T-SQL Operators

- Some operators used in T-SQL expressions:

Arithmetic Operators
Logical Operators
Comparison Operators

- Arithmetic Operators

Operator	Meaning
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Module

T-SQL Operators

- Logical Operators

Operator	Meaning
AND	TRUE if both Boolean expressions are TRUE
BETWEEN	TRUE if the operand is within a range
IN	TRUE if the operand is equal to one of a list of expressions
LIKE	TRUE if the operand matches a pattern
NOT	Reverses the value of any other Boolean operator
OR	TRUE if either Boolean expression is TRUE

- Comparison Operators

Operator	Meaning
=	Equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
<>	Not equal to
!=	Not equal to
!<	Not less than
!>	Not greater than

SELECT...FROM

SELECT select_list
FROM table_source

```sql
-- SELECT simples
SELECT [DepartmentID], [Name], [GroupName]
FROM [AdventureWorks2012].[HumanResources].[Department];
```

100 %

Results | Messages

	DepartmentID	Name	GroupName
1	1	Engineering	Research and Development
2	2	Tool Design	Research and Development
3	3	Sales	Sales and Marketing
4	4	Marketing	Sales and Marketing
5	5	Purchasing	Inventory Management
6	6	Research and Development	Research and Development
7	7	Production	Manufacturing
8	8	Production Control	Manufacturing
9	9	Human Resources	Executive General and Administration
10	10	Finance	Executive General and Administration
11	11	Information Services	Executive General and Administration
12	12	Document Control	Quality Assurance
13	13	Quality Assurance	Quality Assurance
14	14	Facilities and Maintenance	Executive General and Administration
15	15	Shipping and Receiving	Inventory Management
16	16	Executive	Executive General and Administration

SELECT...INTO

SELECT select_list
INTO new_table
FROM table_source

```
5  -- SELECT + INTO
6  SELECT [DepartmentID], [Name], [GroupName] INTO [AdventureWorks2012].[HumanResources].[Department_backup]
7  FROM [AdventureWorks2012].[HumanResources].[Department];
8
```

100 %

Messages

```
(16 row(s) affected)
```

- Tables
  - System Tables
  - FileTables
  - dbo.AWBuildVersion
  - dbo.DatabaseLog
  - dbo.ErrorLog
  - HumanResources.Department
  - HumanResources.Department_backup

SELECT…WHERE

SELECT select_list
FROM table_source
WHERE search_condition

```sql
-- SELECT ... WHERE 1
SELECT [DepartmentID], [Name], [GroupName]
FROM [AdventureWorks2012].[HumanResources].[Department]
WHERE Name = 'Production';

-- SELECT ... WHERE 2
SELECT [DepartmentID], [Name], [GroupName]
FROM [AdventureWorks2012].[HumanResources].[Department]
WHERE Name LIKE 'Production%';
```

100 %

Results | Messages

	Departmen...	Name	GroupName
1	7	Production	Manufacturing

	Departmen...	Name	GroupName
1	7	Production	Manufacturing
2	8	Production Control	Manufacturing

Predicate LIKE

Find strings not exact equal to
Uses wildcards: % and _

SELECT...GROUP BY

SELECT select_list
FROM table_source
GROUP BY group_by_expression

```sql
-- SELECT ... GROUP BY 1 --> ERRO PORQUE QUANDO SE AGRUPA TEM DE SE AGREGAR VALORES
SELECT [DepartmentID], [Name], [GroupName]
FROM [AdventureWorks2012].[HumanResources].[Department]
GROUP BY [GroupName];
```

100 %

Messages

```
Msg 8120, Level 16, State 1, Line 2
Column 'AdventureWorks2012.HumanResources.Department.DepartmentID' is invalid in the select list becaus
```

```sql
-- SELECT ... GROUP BY 2
SELECT [GroupName], [Name]
FROM [AdventureWorks2012].[HumanResources].[Department]
GROUP BY [GroupName], [Name];
```

100 %

Results | Messages

	GroupName	Name
1	Research and Development	Engineering
2	Research and Development	Tool Design
3	Sales and Marketing	Sales
4	Sales and Marketing	Marketing
5	Inventory Management	Purchasing
6	Research and Development	Research and Development

```sql
-- SELECT ... GROUP BY 3
SELECT [GroupName], COUNT([GroupName]) AS Contagem
FROM [AdventureWorks2012].[HumanResources].[Department]
GROUP BY [GroupName];
```

100 %

Results | Messages

	GroupName	Contagem
1	Executive General and Administration	5
2	Inventory Management	2
3	Manufacturing	2
4	Quality Assurance	2
5	Research and Development	3
6	Sales and Marketing	2

SELECT...HAVING

SELECT select_list
FROM table_source
GROUP BY group_by_expression
HAVING search_condition

```sql
-- SELECT ... GROUP BY ... HAVING
SELECT [GroupName], COUNT([GroupName]) AS Contagem
FROM [AdventureWorks2012].[HumanResources].[Department]
GROUP BY [GroupName]
HAVING COUNT([GroupName]) > 2;
```

100 %

Results | Messages

	GroupName	Contagem
1	Executive General and Administration	5
2	Research and Development	3

SELECT...ORDER BY

SELECT select_list
FROM table_source
ORDER BY order_expression [ASC|DESC]

```sql
-- SELECT ... ORDER BY v1
SELECT [DepartmentID], [Name], [GroupName]
FROM [AdventureWorks2012].[HumanResources].[Depart
ORDER BY [Name];

-- SELECT ... ORDER BY v2
SELECT [DepartmentID], [Name], [GroupName]
FROM [AdventureWorks2012].[HumanResources].[Depart
ORDER BY [Name] DESC;
```

100 %

Results | Messages

	DepartmentID	Name	GroupName
1	12	Document Control	Quality Assurance
2	1	Engineering	Research and Developmen
3	16	Executive	Executive General and Adm

	DepartmentID	Name	GroupName
1	2	Tool Design	Research and Developmen
2	15	Shipping and Receiving	Inventory Management
3	3	Sales	Sales and Marketing

```sql
-- SELECT ... ORDER BY v3
SELECT [GroupName], COUNT([GroupName]) AS Contagem
FROM [AdventureWorks2012].[HumanResources].[Departm
GROUP BY [GroupName]
ORDER BY COUNT([GroupName]) DESC;

-- SELECT ... ORDER BY v4
SELECT [GroupName], COUNT([GroupName]) AS Contagem
FROM [AdventureWorks2012].[HumanResources].[Departm
GROUP BY [GroupName]
ORDER BY COUNT([GroupName]) DESC, [GroupName] ASC;
```

100 %

Results | Messages

	GroupName	Contagem
1	Executive General and Administration	5
2	Research and Development	3
3	Sales and Marketing	2
4	Inventory Management	2
5	Manufacturing	2
6	Quality Assurance	2

	GroupName	Contagem
1	Executive General and Administration	5
2	Research and Development	3
3	Inventory Management	2
4	Manufacturing	2
5	Quality Assurance	2
6	Sales and Marketing	2

Select Data with Multiple Tables

- JOIN Types
- INNER JOIN
- CROSS JOIN
- OUTER JOIN
- UNION
- EXCEPT
- INTERSECT

Join Types

- The JOIN clause allows you to combine related data from multiple table sources
- Types of JOIN statements:

Join Type	Description
INNER	Starts with Cartesian product and then applies filter to match rows between tables based on predicate
CROSS	Combines all rows in both tables, aka, Cartesian Product
OUTER [LEFT | RIGHT]	Starts with Cartesian Product and all rows from designated table preserved, matching rows from other table retrieved. Additional NULLs inserted as placeholders

INNER JOIN

- List tables in FROM Clause separated by JOIN operator
- Table aliases preferred
- Table order does not matter

```
SELECT c.CategoryName, p.ProductName
FROM dbo.Categories AS c
INNER JOIN dbo.Products AS p
ON c.CategoryID = p.CategoryID
```

CROSS JOIN

- Combine all possible combinations of two sets, aka, Cartesian Product

Name
Davis
Funk
King

×

Product
Alice Mutton
Crab Meat
Ipoh Coffee

=

Name	Product
Davis	Alice Mutton
Davis	Crab Meat
Davis	Ipoh Coffee
Funk	Alice Mutton
Funk	Crab Meat
Funk	Ipoh Coffee
King	Alice Mutton
King	Crab Meat
King	Ipoh Coffee

```
SELECT c.CategoryName, p.ProductName
FROM dbo.Categories AS c
CROSS JOIN dbo.Products AS p
```

OUTER JOIN

- LEFT OUTER JOIN
  - Return all rows from first table and only matches from second

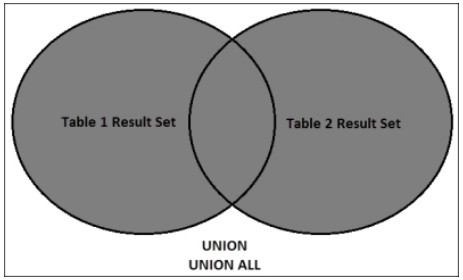
```
SELECT c.ContactName, o.OrderID
FROM dbo.Customers AS c
LEFT OUTER JOIN dbo.Orders AS o
ON c.CustomerID = o.CustomerID
```

- RIGHT OUTER JOIN
  - Return all rows from second table and only matches from first

```
SELECT c.ContactName, o.OrderID
FROM dbo.Customers AS c
RIGHT OUTER JOIN dbo.Orders AS o
ON c.CustomerID = o.CustomerID
```

UNION

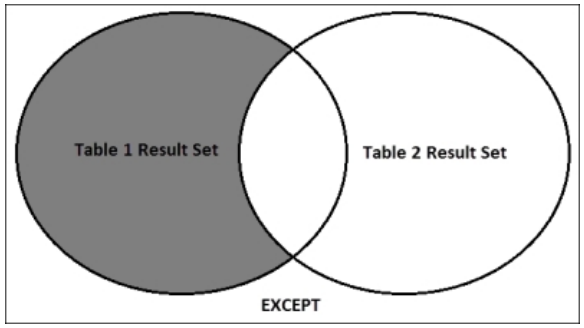
- Used to combine sets from the same table or different tables
- All selected columns need to be of the same data type




```
-- List the countries where we have customers or
suppliers

-- Without union
SELECT Country FROM dbo.Suppliers
SELECT Country FROM dbo.Customers

-- With union
SELECT Country FROM dbo.Suppliers
UNION
SELECT Country FROM dbo.Customers
```

EXCEPT

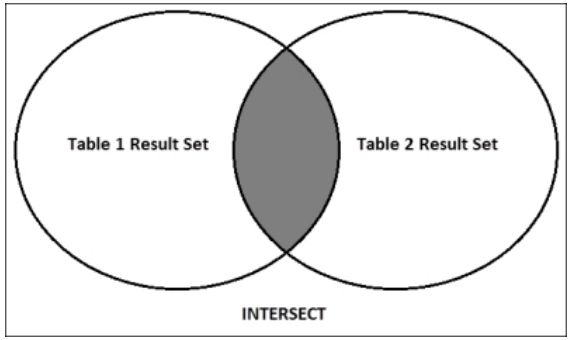
- Used to find what records exists in one set and not in the other set




```
-- Return a list of customers countries where there are no suppliers
SELECT DISTINCT Country FROM dbo.Customers
EXCEPT
SELECT DISTINCT Country FROM dbo.Suppliers
```

INTERSECT

- Used to find common values between sets from different tables




```
-- Create a list with countries from both our customers and suppliers
SELECT DISTINCT Country FROM dbo.Suppliers
INTERSECT
SELECT DISTINCT  Country FROM dbo.Customers
```

Insert, Update and Delete Data

- INSERT Statement
- UPDATE Statement
- DELETE Statement

INSERT Statement

- This statement inserts a single row by default

```sql
-- Omitting the PK
INSERT INTO dbo.Categories
(
  CategoryName,
  [Description]
)
VALUES
(
  'New Category',
  'With nothing...'
);
```

```sql
-- Insert value into PK
INSERT INTO dbo.Region
(
  RegionID,
  RegionDescription
)
VALUES
(
  (SELECT MAX(RegionID) FROM dbo.Region ) + 1,
  'New Region'
);
```

UPDATE Statement

- UPDATE statement is used to modify data
  - Updates rows in a table or view filtered with a WHERE clause
  - Only columns specified in the SET clause are modified

```
UPDATE dbo.Region
SET RegionDescription = 'Old Region'
WHERE RegionDescription =  'New Region';
```

```
UPDATE dbo.Products
SET UnitPrice = (UnitPrice * 1.10), Discontinued = 0
WHERE ProductName =  'Guaraná Fantástica';
```

```
-- Suggestion: add a table Notes field and set it’s value to ‘changed’ to rollback
UPDATE dbo.Products
SET Discontinued = 0
FROM dbo.Categories AS c
INNER JOIN dbo.Products AS p
ON c.CategoryID = p.CategoryID
WHERE p.Discontinued = 1
```

DELETE Statement

- DELETE without a WHERE clause deletes all row
  - In this case is better to use TRUNCATE TABLE
    - Minimally logged
    - It will fail if the table is referenced by a foreign key constraint in another table
- Use a WHERE clause to delete specific rows

```sql
DELETE FROM dbo.Region;
```

```sql
DELETE FROM dbo.Region
WHERE RegionDescription = 'Old Region';
```

```sql
DELETE dbo.Products
FROM dbo.Categories AS c
INNER JOIN dbo.Products AS p
ON c.CategoryID = p.CategoryID
WHERE c.CategoryName = 'New Category';
```

Demo

- Using Northwind and execute several DML statements to train your new skills.

Practice

- See exercise.

Summary

- Use DML statements to extract and manipulate data in SQL Server database tables
- The DML core statement is SELECT that is used to query data
- When you need to query multiple tables you must use JOINS
- The other DML (INSERT, UPDATE, DELETE) statements achieves changing data, which is the daily basis of transactional databases

Administrating Databases

Module 7

Overview

1. Securing SQL Server
2. Backing Up and Restoring Databases
3. Demo
4. Practice
5. Summary

Securing SQL Server

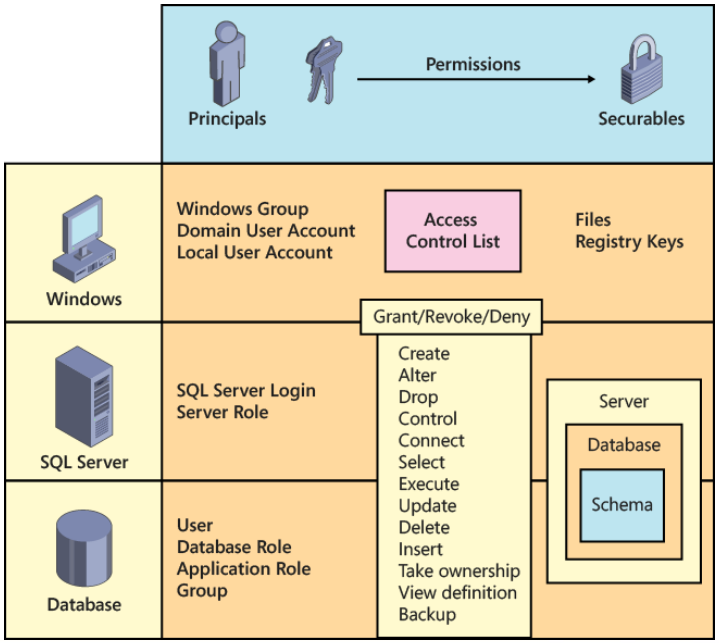
- Authentication and Authorization
- SQL Server Security
- Fixed Server Roles
- Fixed Database Roles
- Permissions

Authentication and Authorization

- Process
  - First, a user must be authenticated in SQL Server
  - Second, authenticated users have access to server/database objects
  - Third, authenticated users may or may not have permissions to manipulate objects
- Authentication
  - Is the act of establishing or confirming a user or system identity
- Authorization
  - We assign permissions to objects through logins
  - We use GRANT, DENY or REVOKE commands

SQL Server Security

- Principals are security identities that access securables and perform actions, for example, a server login or a database user
- Securables are objects to which access must be secured, for example, a table or a stored procedure
- Permissions are the actions principals can perform on securables



SQL Server Security

- Logins and Accounts
  - A login/logon is the identification process of the user's credentials when accessing a computer/server/database system
  - A combination of username and password is the most common login method
- Server-Level Security
  - Methods for which a user can be identified:
    - Membership in a Windows user group
    - Windows AD user login
    - SQL Server-specific login
- Database-Level Security
  - All users are automatically members of the public database role
  - Users belong to database roles from which they inherit permissions

Fixed Server Roles

- We can think of roles as groups with permissions by default
- SQL Server includes fixed server roles:
  - Bulkadmin
  - Dbcreator
  - Diskadmin
  - Processadmin
  - Securityadmin
  - Serveradmin
  - Setupadmin
  - Sysadmin

Fixed Database Roles

- SQL Server includes fixed database roles:
  - db_accessadmin
  - db_backupoperator
  - db_datareader
  - db_datawriter
  - db_ddladmin
  - db_denydatareader
  - db_denydatawriter
  - db_owner
  - db_securityadmin

Permissions

- Object permissions allows a user to act on server/database objects, such as tables, stored procedures, and views:
  - Select
  - Insert
  - Update
  - Delete
  - Execute
- The permissions are inherited from roles or managed individually
- Ownership chaining enables managing access to multiple objects, such as multiple tables, by setting permissions on one object, such as a view

Backing Up and Restoring Databases

- Recovery Model
- Backup and Restore
- Backup
- Restore

Recovery Model

- Transaction log
  - Provides a history of actions executed by a database management system to guarantee atomicity and durability of transactions
- Recovery model
  - Defines if the database uses transaction log
  - Database > Properties > Options > Recovery Model

Recovery Model	Description
Simple	Does not permit or require log backups Automatically truncates log to keep space requirements small
Full	Requires log backups for manageability Avoids data loss due to a damaged or missing data file Permits recovery to a specified point in time
Bulk Logged	Requires log backups for manageability Can enhance the performance of bulk copy operations Reduces log space usage by using minimal logging for many bulk operations

Backup and Restore

- Backup
  - Saves data for future need or recover from disaster situations
  - SQL Server backups are created on backup devices, such as disk files or tape media
  - You can append new backups to any existing backups on a device or overwrite any existing backups
- Restore
  - Is the process of restoring backups taken using the BACKUP command
  - The type of restore depends of the type of backup

Backup

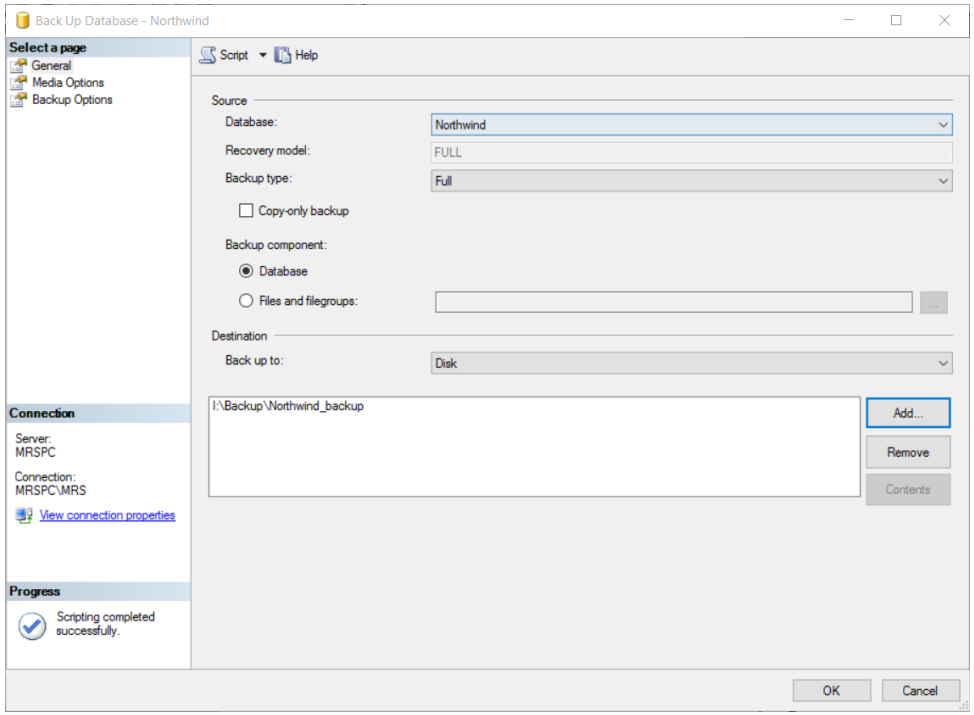
- Types of backups:

Backup type	Description
Full	All data files and the active part of the transaction log
Differential	The parts of the database that have changed since the last full database backup
Transaction Log	Any database changes recorded in the log files
File/File Group	Specified files or filegroups
Partial	The primary filegroup, every read/write filegroup, and any specified read-only filegroups
Tail-log Backup	Log backup taken of the tail of the log just before a restore operation
Copy Only	The database or log (without affecting the backup sequence)

Backup

```
BACKUP DATABASE Northwind
TO  DISK = N'I:\Backup\Northwind_backup.bak';
```

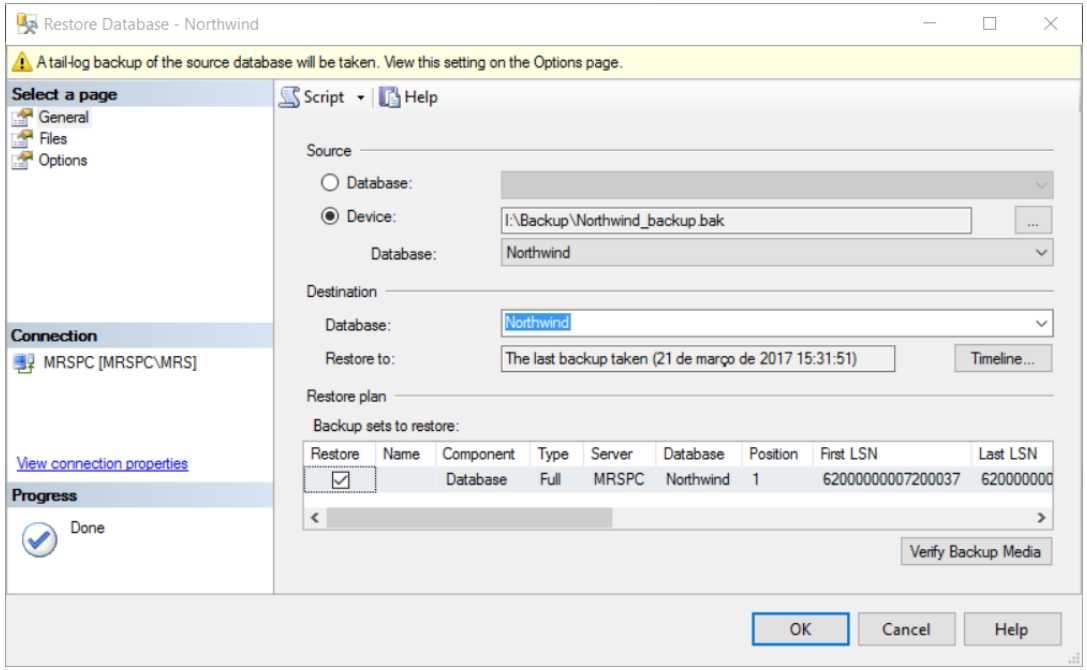
- Object Explorer > Database > Right-click > Tasks > Back Up



Restore

```
RESTORE DATABASE Northwind
FROM DISK = 'I:\Backup\Northwind_backup.bak';
```

- Object Explorer > Databases > Right-click > Restore Database



Demo

- Using Northwind, ensure that you have a full backup saved to D:\Backup.
- Delete Northwind database.
- Restore Northwind database from previous backup.

Practice

- See exercise.

Summary

- In order to manipulate objects in a database, we must ensure that SQL Server Security is properly defined. It involves principals, securables, authorization and authentication.
- Users have permissions inherited from the roles they belong to or granted directly.
- Permissions can be granted, denied or revoked.
- Each database project must include a correct recovery model mode as well as a backup/restore policy.