

## Relational Data

- In a relational database, you model collections of entities from the real world as *tables*.
- Data is represented in Tabular format in rows and columns.
- Relational tables are a format for structured data
- Each Row (record) represents an Entity.
- Each row has same set of columns.
- Each column describes **attributes** of Entity and has datatype associated with it.
- A relational database allows the definition of data structures, storage and retrieval operations and integrity constraints.
- The most popular RDBMS are MS SQL Server, DB2, Oracle and MySQL.

### Example:

Employee			Department		
ID	Name	Salary	ID	Name	Location
1001	John	10000	101	Sales	Block A
1002	Allen	15000	102	IT	Block B
1003	Smith	18000			

### Properties of Relational Tables:

- Values Are Atomic
- Each Row is Unique
- Column Values are of the Same Kind.
- The Sequence of Columns is Insignificant
- The Sequence of Rows is Insignificant
- Each Column Has a Unique Name

### E-R Modeling:

E-R Modeling is a technique of data modeling that is useful in developing a conceptual design for a database.

The whole purpose of ER modeling is to create an accurate reflection of the real world in a database.

### Entity:

- It is a real world object which is distinguishable from any other object.
- An entity is a representation of an item which can be physical (such as a customer or a product), or virtual (such as an order).

- Entity is a person, place, object, event or concept in the user environment about which the organization wishes to maintain Data.
- Entity can be concrete or abstract
- Concrete entity Example : Specific person, Company, Course
- Abstract entity Example: Specific Order, Holiday

#### **Attributes:**

- A named property or characteristic of an entity that is of interest to an organization
- Example: People have names and addresses.

#### **Entity Type:**

- It is collection of same types of entities having same attributes.
- Each Entity set has a key.
- Example: people, company, course, order, transaction.

#### **Entity set**

- Collection of Entities which contain entities of particular entity type
- It is referred using the same name as entity type.

#### **Relationship:**

- It is association between entities enabling interaction
- representation of the fact that certain entities are related to each other.
- Example: Customer buys a product

Artist performs a song.

Students register in courses

Passengers books flights

Bank Has branches,

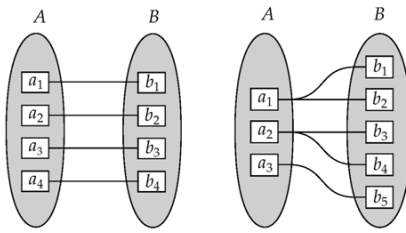
Customers have accounts

#### **Mapping Cardinalities:**

- Relationship is expressed in terms of cardinality.
- Cardinality gives number of entities in one entity set, to which another entity in another entity set can be associated via a relationship.

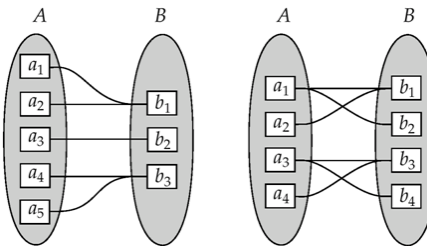
#### **Relation:**

- Entities are connected by relations enabling interaction.
- For a binary relationship set the mapping cardinality must be one of the following types:
- One to One, One to Many, Many to One and Many to Many



One to One (Department – Address)

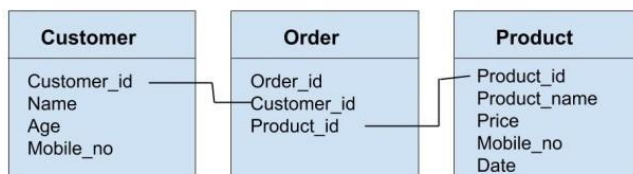
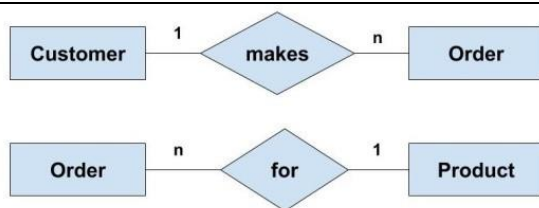
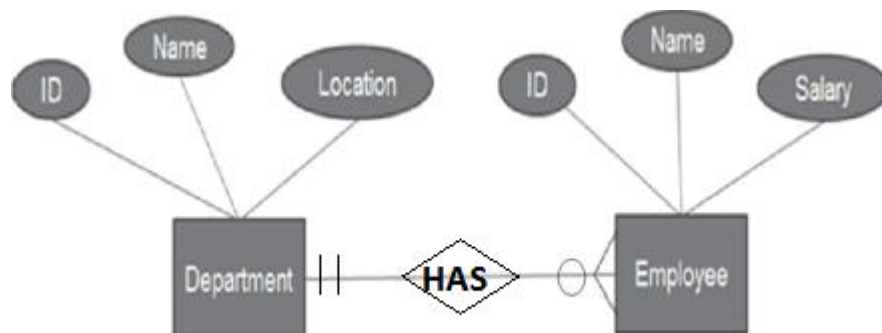
One to Many (Department – Employee)

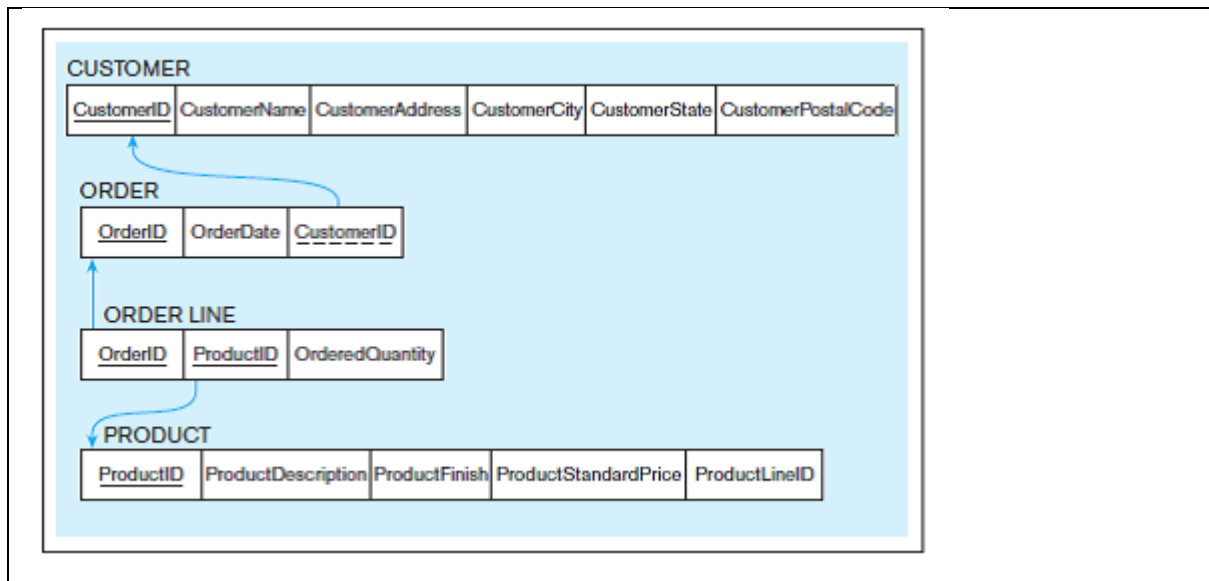


Many to One (Order – Customer)

Many to Many (Faculty – Course)

### Example: Example of E-R Diagram





### Relationships between Entities in RDBMS:

Relationship is created in tables using primary and foreign key.

**Primary Key:** A field (or fields) used to identify an Entity Uniquely.

**Foreign Key:** A field (or fields) on the many side of a one-to-many relationship between tables that relates to the primary key of the other table.



ID	Name	Location
101	Sales	Block A
102	IT	Block B

ID	Name	Salary	Department
1001	John	10000	101
1002	Allen	7000	102
1003	Smith	15000	101

### Normalization:

- Normalization is a database design technique, a systematic approach of decomposing tables to eliminate data redundancy(repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies.

- It is a multi-step process based on rules called as normal forms, which puts data into separate specific table, establishing required relationship between the tables
- A simple definition of Normalization for practical purpose is
- Separate each *entity* into its own table.
- Separate each discrete *attribute* into its own column.
- Uniquely identify each entity instance (row) using a *primary key*.
- Use *foreign key* columns to link related entities.
- Normalization is used for mainly two purposes
  - Eliminating redundant data.
  - Ensuring data dependencies make sense i.e data is logically stored.
  - Result of the normalization process is that your data is split into many narrow, well-defined tables (a *narrow* table is a table with few columns)
- **In a Normalized Database schema:**
  - Primary Keys and Foreign keys are used to define relationships and enforce referential integrity
  - No data duplication exists
  - Data is retrieved by joining tables together in a query.

## SQL

SQL is a standard language for use with relational databases

Standards are maintained by ANSI and ISO

Most RDBMS systems support proprietary extensions of standard SQL

Some popular dialects of SQL include:

- *Transact-SQL (T-SQL)*. This version of SQL is used by Microsoft SQL Server and Azure SQL services.
- *pgSQL*. This is the dialect, with extensions implemented in PostgreSQL.
- *PL/SQL*. This is the dialect used by Oracle. PL/SQL stands for Procedural Language/SQL.

## Types of SQL Statements

1. **Data Definition Language (DDL)** : Create, Alter, Drop, Truncate

It is used to create and modify the structure of database objects in database.

2. **Data Manipulation Language (DML)** : Insert, Update, Delete, [Select]

It is used to retrieve, store, modify, delete, insert and update data in database.

3. **Data Query Language (DQL)** : Select

It is used to retrieve data in the database.

4. **Transaction Control Language (TCL)** : Commit, Rollback, SavePoint

It is used to manage different transactions occurring within a database.

5. **Data Control Language (DCL)** : Grant, Revoke

It is used to create roles, permissions to control access to database by securing it.

**Examples:**

Refer following Scripts:

01\_Sample Tables.sql

02\_Sample T-SQL Queries

## Common Database Objects

### What is View?

- View is virtual table based on result set of query.
- Views are useful:
  - To simplify query
  - Combine relational data in single pane view.

### What is Stored Procedure?

- Stored procedure contains pre -defines SQL statements that can be run on command.
- Stored procedures are used to encapsulate programmatic logic in a database for actions that applications need to perform when working with data.
- It can contain Parameters.

### What is Index?

- Optimizes search queries for faster data retrieval
- Reduces the amount of data pages that need to be read to retrieve the data in a SQL Statement

### Example:

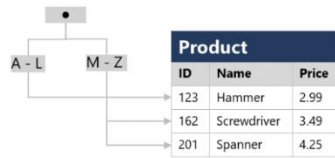
**Refer: 03\_Database Objects.sql**

Create Index on Name Column in Product Table.

```
CREATE INDEX idx_ProductName
```

```
ON Product(Name);
```

The index creates a tree-based structure that the database system's query optimizer can use to quickly find rows in the **Product** table based on a specified **Name**.



## Relational Database Services in Azure

Azure supports multiple database services, enabling you to run popular relational database management systems, such as SQL Server, PostgreSQL, and MySQL, in the cloud.

### Azure SQL:

Azure SQL is a collective term for a family of Microsoft SQL Server based database services in Azure.

It has following Implementaions:

#### 1. SQL Server on Azure Virtual Machines (VMs) :

- A virtual machine running in Azure with an installation of SQL Server.
- Guaranteed compatibility to SQL Server on premises
- Customer manages everything – OS upgrades, software upgrades, backups, replication
- Pay for the server VM running costs and software licensing, not per database
- Great for hybrid cloud or migrating complex on-premises database configurations(Lift and Shift)

#### 2. Azure SQL Managed Instance:

- A platform-as-a-service (PaaS) option that provides near-100% compatibility with on-premises SQL Server instances
- Automated software update management, backups, and other maintenance tasks, reducing the administrative burden of supporting a database server instance.
- Use a single instance with multiple databases, or multiple instances in a pool with shared resources
- Great for migrating most on-premises databases to the cloud

#### 3. Azure SQL Database:




- Core database functionality compatibility with SQL Server
- Automatic backups, software patching, database monitoring, and other maintenance tasks
- Single database or elastic pool to dynamically share resources across multiple databases
- Azure SQL Database is often used for:

Modern cloud applications that need to use the latest stable SQL Server features.

Applications that require high availability.

Systems with a variable load that need the database server to scale up and down quickly.

## Comparison

Azure SQL Database (Logical server) 	SQL Managed Instance 	SQL Server on VM 
PAAS Service	PAAS Service	IAAS Service
The most commonly used SQL Server features are available.	Near-100% compatibility with SQL Server. on-premises.	Fully compatible with on-premises physical and virtualized installations.
You can provision a <i>single database</i> in a dedicated, managed (logical) server; or you can use an <i>elastic pool</i> to share resources across multiple databases and take advantage of on-demand scalability.	Each managed instance can support multiple databases. Additionally, <i>instance pools</i> can be used to share resources efficiently across smaller instances.	SQL Server instances are installed in a virtual machine. Each instance can support multiple databases.
99.995% availability guaranteed.	99.99% availability guaranteed.	Up to 99.99% availability.
Latest stable Database Engine version.	Latest stable Database Engine version.	Fixed, well-known database engine version.
Fully automated updates, backups, and recovery.	Fully automated updates, backups, and recovery.	You must manage all aspects of the server, including operating system and SQL Server updates, configuration, backups, and other maintenance tasks.
Migration from SQL Server might be hard.	Easy migration from SQL Server.	Easy migration from SQL Server on-premises.

## Azure Database services for open-source

Azure data services are available for other popular relational database systems, including MySQL, MariaDB, and PostgreSQL.

### MySQL:

- Leading open source relational database for Linux, Apache, MySQL, and PHP (LAMP) stack apps



### **MariaDB:**

- Created by the original developers of MySQL
- MariaDB offers compatibility with Oracle Database (another popular commercial database management system).
- One notable feature of MariaDB is its built-in support for temporal data. A table can hold several versions of data, enabling an application to query the data as it appeared at some point in the past.

### **PostgreSQL:**

- PostgreSQL is a hybrid relational Object database.
- You can store data in relational tables, but a PostgreSQL database also enables you to store custom data types, with their own non-relational properties.
- Another key feature is the ability to store and manipulate geometric data, such as lines, circles, and polygons.
- PostgreSQL has its own query language called *pgsql*

### **Azure Database for MySQL:**

- PaaS implementation of MySQL based on community edition
- Azure Database for MySQL has two deployment options: **Single Server** and **Flexible Server**.
- Both options are fully managed database as a service offering, with predictable performance and dynamic scalability.
- **Flexible Server** provides more granular control and flexibility over database management functions and configuration settings
- Flexible server is the **recommended** deployment option for **all new developments or migrations**.
- **Single servers** are best for **existing applications** already using single server.

### **Azure Database for MariaDB:**

- An implementation of the MariaDB Community Edition database management system adapted to run in Azure
- Compatibility with Oracle Database
- The database is fully managed and controlled by Azure and almost no additional administration is needed

### **Azure Database for PostgreSQL**

- Database service in the Microsoft cloud based on the PostgreSQL Community Edition database engine
- Hybrid relational and object storage
- Deployment options: Single Server, Flexible Server, Hyperscale
- **Single Server** has Three pricing tiers: Basic, General Purpose, and Memory Optimized.
- Each tier supports different numbers of CPUs, memory, and storage sizes; you select one based on the load you expect to support.

- **Flexible-server** deployment option for PostgreSQL provides more control and server configuration customizations, and has better cost optimization controls.
- **Hyperscale** (Citus) is a deployment option that scales queries across **multiple server nodes** to support large database loads.
- Data is split into chunks based on the value of a partition key

Demo:

[Exercise: Provision Azure relational database services - Learn | Microsoft Docs](#)