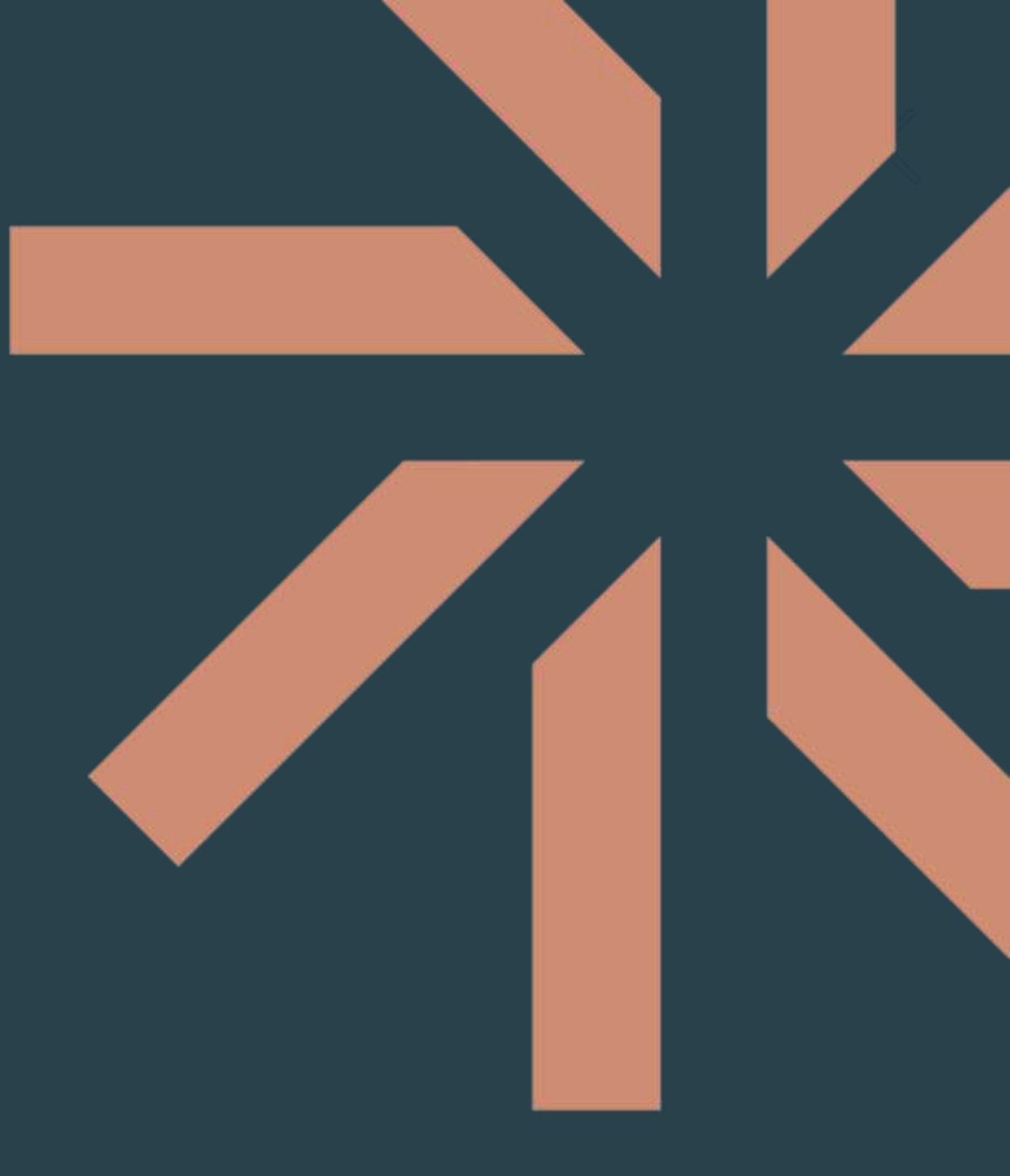
Database Development and Design

Qinshift Academy

Database Development and Design

Developing and Design of databases using PostgreSQL - Powerful, open-source object-relational database



Major course intro

- Understanding the Core Database Concepts
  - Database Server, Database, Data Types, DDL, DML
  - INSERT, UPDATE, DELETE
- Querying data
  - Select statements, Filtering, Sorting, Unions, Joins
  - Foreign keys
- Advanced querying
  - Aggregate functions, Grouping data, Views
- Database routines
  - Stored procedures, Functions
- Error handling and Basic Administration
  - Triggers and Constraints, Error handling
  - Users, Permissions, Roles

Agenda

- Session 1
  - Database concepts and usage
  - Types of databases
  - Introduction to pgAdmin 4
  - Key terminology (DDL and DML statements)
    - Demo 1 - DDL and DML
  - Data types
    - Demo 2 – Data types
  - CRUD Operations (Create, Read, Update, Delete)
    - Create, Insert, Update, Delete
  - Workshop – Creating tables
  - Knowledge check (Quiz, Discussion, Homework)
- Session 2
- Session 3
- Session 4
- Session 5
- Session 6
- Session 7

Database concepts and usage

Database definitions 1/2

Cambridge dictionary

A large amount of information stored in a computer system in such a way that it can be easily looked at or changed.

Wikipedia

Database is an organized collection of data, generally stored and accessed electronically from a computer system. Where databases are more complex, they are often developed using formal design and modeling techniques.

Britannica

Database, also called electronic database, any collection of data, or information, that is specially organized for rapid search and retrieval by a computer. Databases are structured to facilitate the storage, retrieval, modification, and deletion of data in conjunction with various data-processing operations.

Database definitions 2/2

- A database consists of logically related data stored in a single repository.

- Provides advantages over file system management approach
  - Eliminates inconsistency, data anomalies, data dependency, and structural dependency problems
  - It stores data structures, relationships, and access paths

Example of an old traditional database application

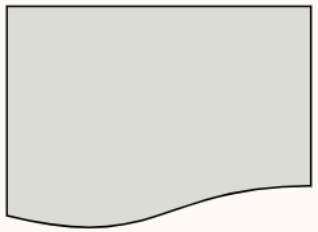
Suppose we are building a system to store the information about:

- Customers
- Products
- Orders

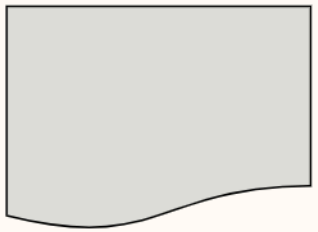
Can we do it without a database management system (DBMS)?

We can if we store the data in files

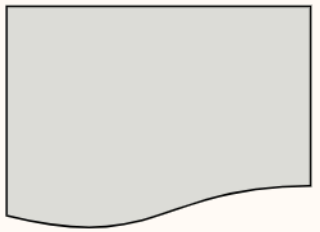
Customer.txt



Product.txt



Order.txt



Database management system (DBMS)

- A database management system (DBMS) is a software package designed to define, manipulate, retrieve and manage data in a database.
- A DBMS generally manipulates the data itself, the data format, field names, record structure and file structure.
- It also defines rules to validate and manipulate this data.

Doing it without DBMS

- Suppose we need to create a program to execute specific tasks with the data about customers, products and orders
- Perform an order: “Customer A” orders “Product A”
  - Write a program to do the following:

Read ‘Customer.txt’
Read ‘Product.txt’
Write in ‘Orders.txt’ - “Customer A orders Product A”
Write in “Product.txt” - “Product A” has changed available quantity

Problems without DBMS

- Large data sets (Millions of Customers and Products)
  - Creates problem with memory
- Concurrent access by many users
  - Multiple users want to place an order in a same time. Multiple requests for write to Orders data file.
- Data inconsistency
  - Data inconsistency occurs when different versions of the same data exist in different places in an organization.
- A management system helps get quick solutions to database queries, thus making data access faster and more accurate. End-users like salespeople will have enhanced access to the data, enabling more productivity.

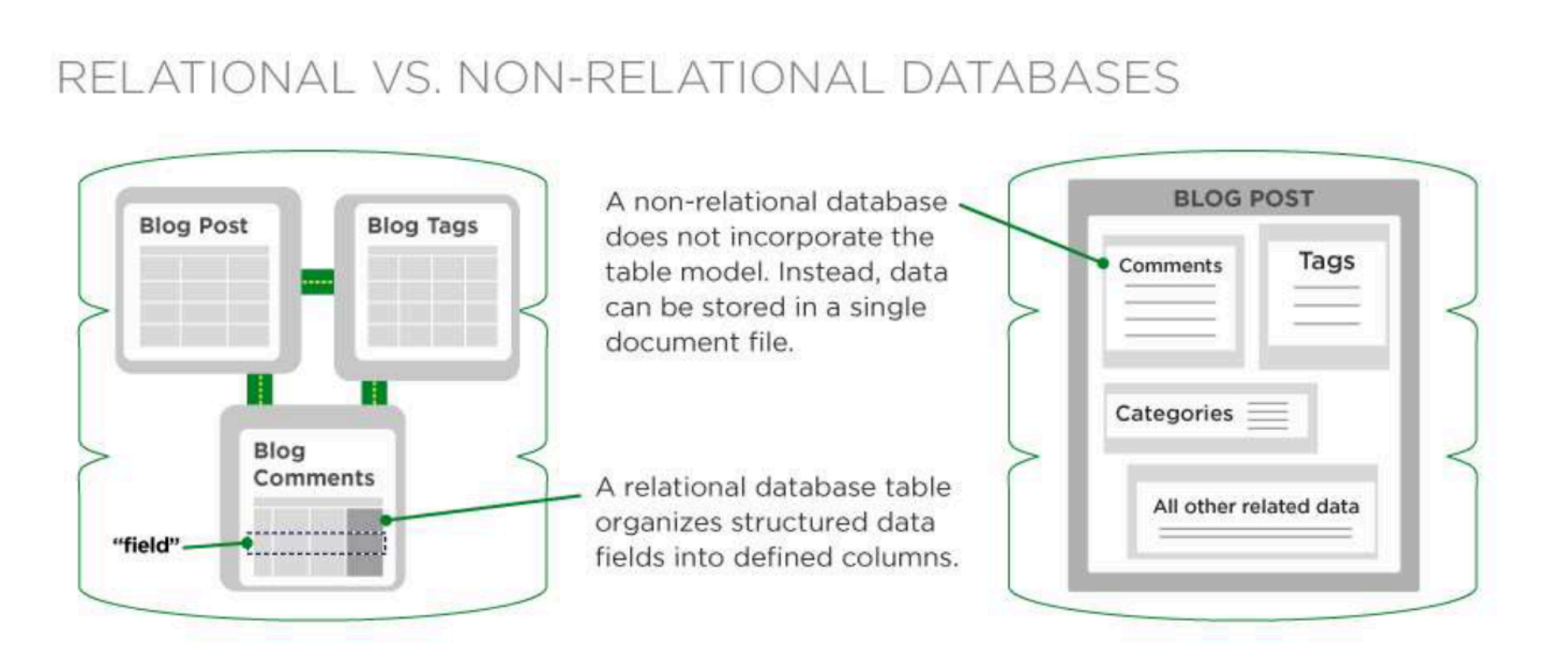
PostgreSQL

- PostgreSQL is an advanced, enterprise-class, and open-source relational database system. PostgreSQL supports both SQL (relational) and JSON (non-relational) querying.
- PostgreSQL is a highly stable database backed by more than 20 years of development by the open-source community.
- PostgreSQL is used as a primary database for many web applications as well as mobile and analytics applications.

Types of databases

Types of databases 1/3

- Relational databases (SQL)
- Non-Relational databases (NoSQL)



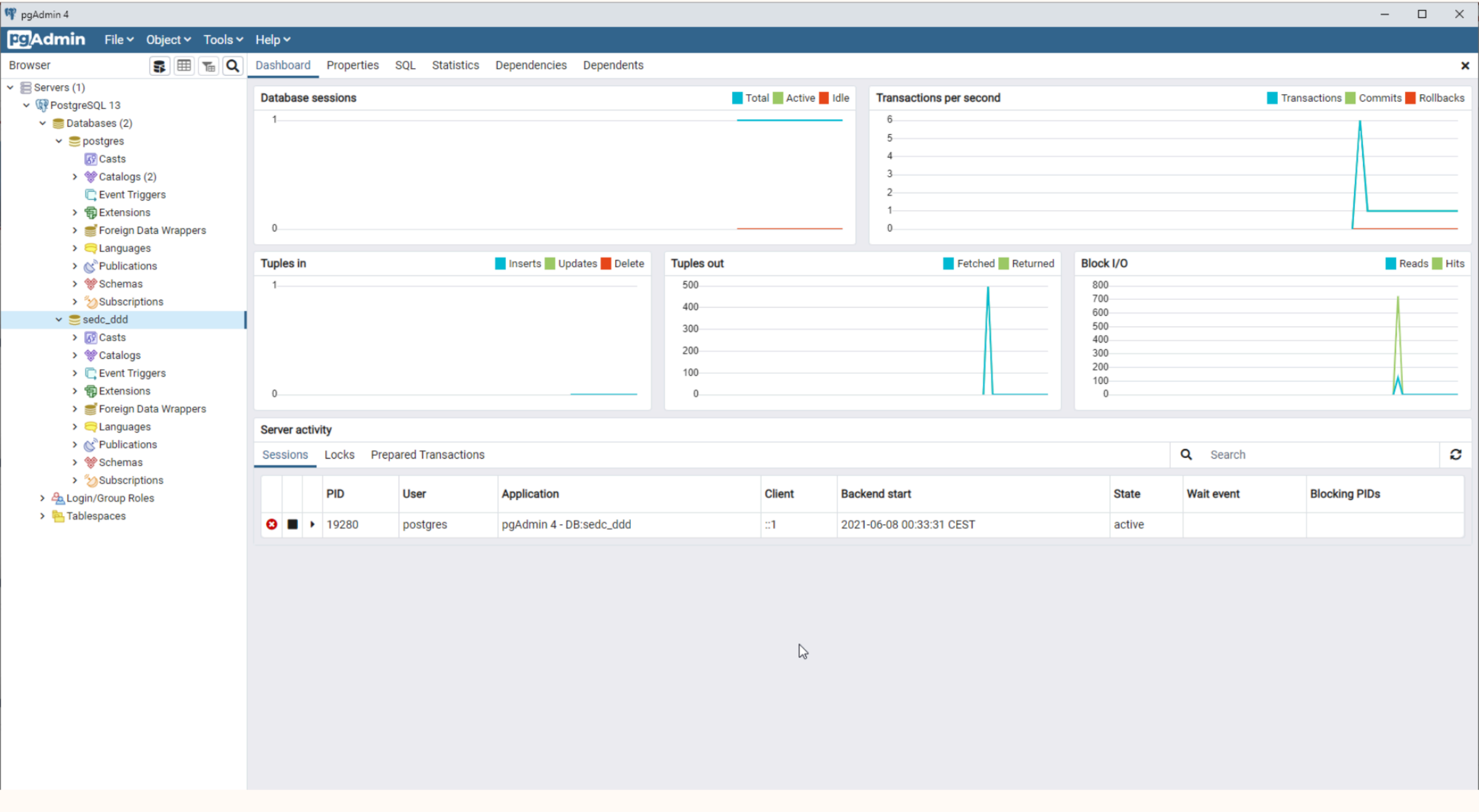
Types of databases 2/3

SQL	NoSQL
RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS)	Non-relational or distributed database system
These databases have fixed or static or predefined schema	They have dynamic schema
These databases are not suited for hierarchical data storage	These databases are best suited for hierarchical data storage.
These databases are best suited for complex queries	These databases are not so good for complex queries
Vertically Scalable	Horizontally scalable

Types of databases 3/3

- Relational databases (SQL)
  - Structured (table based)
  - Most popular
    - SQL Server, Oracle, PostgreSQL, MySQL
- Non-Relational databases (SQL)
  - Non structured
    - Document based
    - Key-value pairs
    - Graph databases
    - Wide column stores
  - Most popular
    - MongoDB, Cassandra, Hbase, CoachDB

Introduction to pgAdmin 4


pgAdmin 4
pgAdmin
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Servers (1)
PostgreSQL 13
Databases (2)
postgres
Casts
Catalogs (2)
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas
Subscriptions
sedc_ddd
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas
Subscriptions
Login/Group Roles
Tablespaces
Database sessions
Total
Active
Idle
Transactions per second
Transactions
Commits
Rollbacks
Tuples in
Inserts
Updates
Delete
Tuples out
Fetched
Returned
Block I/O
Reads
Hits
Server activity
Sessions
Locks
Prepared Transactions
Search
PID
User
Application
Client
Backend start
State
Wait event
Blocking PIDs
19280
postgres
pgAdmin 4 - DB:sedc_ddd
::1
2021-06-08 00:33:31 CEST
active

Key terminology

Key database terminology 1/2

Data Definition Language (DDL) is a subset of the Transact-SQL language; it deals with creating database objects like tables, constraints, and stored procedures. The interface used to create these underlying DDL statements is the pgAdmin user interface.

The main DDL statements are as follows:

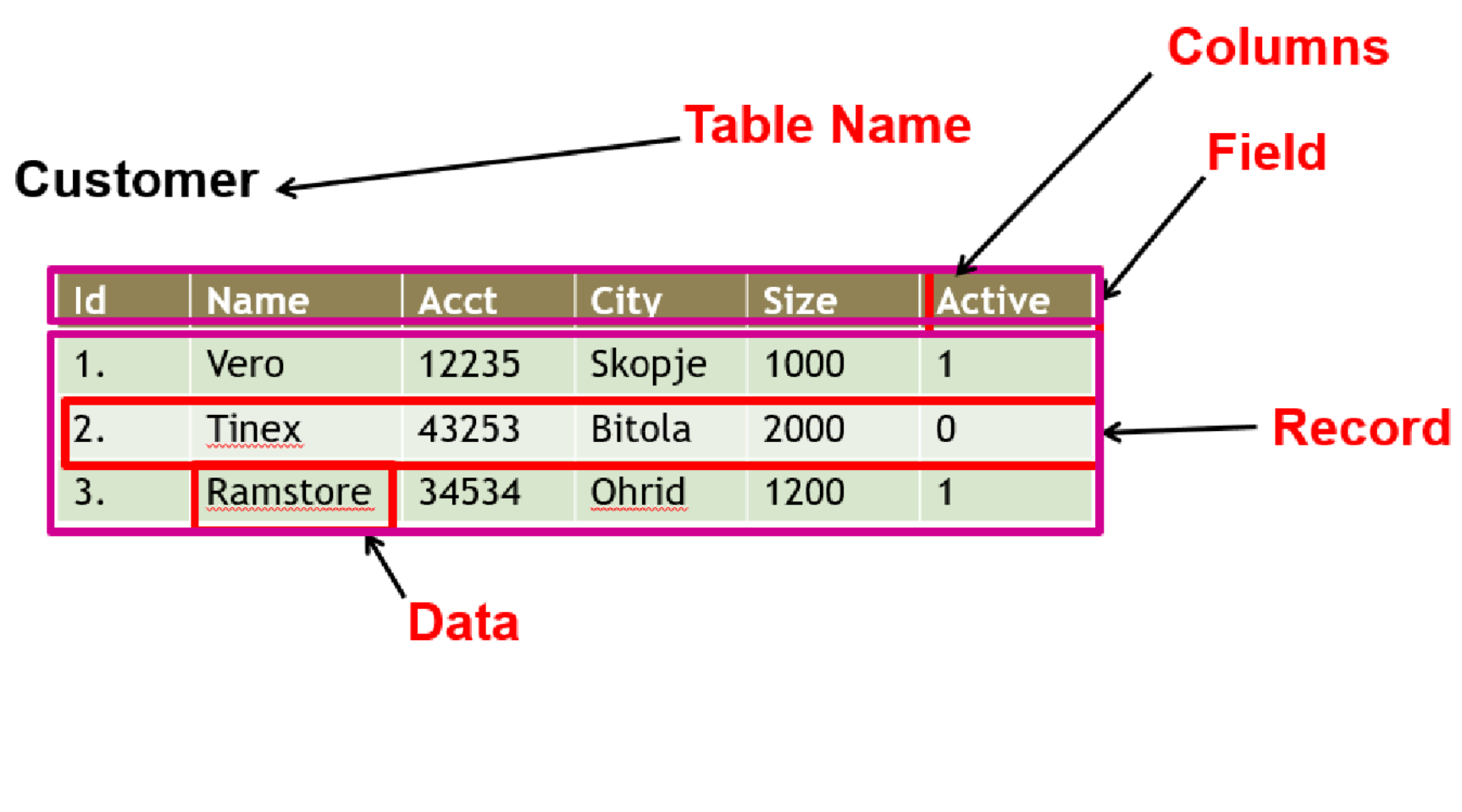
- **ALTER**
- **CREATE**
- **DROP**
- **GRANT**
- **REVOKE**

Key database terminology 2/2

Data Manipulation Language (DML) is the language element that allows you to use the core statements INSERT, UPDATE and DELETE to manipulate data in any SQL Server tables. Core DML statements include the following:

- **SELECT**: Retrieves rows from the database and enables the selection of one or many rows or columns from one or many tables in SQL Server.
- **INSERT**: Adds one or more new rows to a table or a view in SQL Server.
- **UPDATE**: Changes existing data in one or more columns in a table or view.
- **DELETE**: Removes rows from a table or view.

A table example terminology



A table is a collection of related data held in a table format within a database. It consists of columns and rows.

Data types in PostgreSQL

Data types in PostgreSQL

- Each column in a database table is required to have a name and a data type.
- SQL data type is an attribute that specifies type of data of any object. Each column, variable and expression has related data type in SQL.
- You would use these data types while creating your tables. You would choose a particular data type for a table column based on your requirement.
- SQL developers must decide what types of data will be stored inside each table column when creating a SQL table.
- The data type is a label and a guideline for SQL to understand what type of data is expected inside of each column, and it also identifies how SQL will interact with the stored data.
- In PostgreSQL, 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: Boolean, character, numeric, temporal, array, json, uuid, and special types.

Data types in PostgreSQL

- **Boolean**
  - A Boolean data type can hold one of three possible values: true, false or null. You use boolean or bool keyword to declare a column with the Boolean data type.
- **Character**
  - PostgreSQL provides three character data types: CHAR(n), VARCHAR(n), and TEXT
    - **CHAR**(n) is the fixed-length character with space padded. If you insert a string that is shorter than the length of the column, PostgreSQL pads spaces. If you insert a string that is longer than the length of the column, PostgreSQL will issue an error.
    - **VARCHAR**(n) is the variable-length character string. With VARCHAR(n), you can store up to n characters. PostgreSQL does not pad spaces when the stored string is shorter than the length of the column.
    - **TEXT** is the variable-length character string. Theoretically, text data is a character string with unlimited length.

Data types in PostgreSQL

- **Numeric**
  - **Integer**
    - There are three kinds of integers in PostgreSQL:
      - Small integer (**SMALLINT**) is 2-byte signed integer that has a range from -32,768 to 32,767.
      - Integer (**INT**) is a 4-byte integer that has a range from -2,147,483,648 to 2,147,483,647.
      - Serial is the same as integer except that PostgreSQL will automatically generate and populate values into the **SERIAL** column. This is similar to AUTO_INCREMENT column in MySQL or AUTOINCREMENT column in SQLite.
  - **Floating-point number**
    - There three main types of floating-point numbers:
      - **float**(n) is a floating-point number whose precision, at least, n, up to a maximum of 8 bytes.
      - **Real** or **float8** is a 4-byte floating-point number.
      - **numeric** or **numeric(p,s)** is a real number with p digits with s number after the decimal point. The numeric(p,s) is the exact number.

Data types in PostgreSQL

- **Temporal data types**
  - The temporal data types allow you to store date and /or time data. PostgreSQL has five main temporal data types:
    - **DATE** stores the dates only.
    - **TIME** stores the time of day values.
    - **TIMESTAMP** stores both date and time values.
    - **TIMESTAMPTZ** is a time zone-aware timestamp data type. It is the abbreviation for timestamp with the time zone.
    - **INTERVAL** stores periods of time.

Data types in PostgreSQL

- **UUID** for storing Universally Unique Identifiers
- **Array** for storing array strings, numbers, etc.
- **JSON** stores JSON data
- **hstore** stores key-value pair
- Special types such as network address and geometric data.

CRUD operations

Create, Read, Update, Delete

CREATE operation 1/2

- Create new data structure (Table, View, Function, Procedure, …)

```
CREATE TABLE TableName (
    Column1_Name Data_type_1,
    Column2_Name Data_type_2,
    ...
    ColumnN_Name Data_type_N)
```

- Example:

```
CREATE TABLE Customer (
        Id INTEGER NOT NULL,
        Name varchar(100) NOT NULL,
        City varchar(100) NULL)
```

Create operation 2/2

- Primary key concept
- Short, Not changeable, Incremental, Unique

```
CREATE TABLE Customer (
Id INTEGER PRIMARY KEY,
Name TEXT,
Address TEXT);
```

INSERT operation

- Insert new data

```
INSERT INTO TableName (Col1,
Col2, ...)
VALUES (Value1, Value2, ...)
```

- Example:

```
INSERT INTO Customer (Id, Name, City)
VALUES  (1, 'Vero Skopje', 'Skopje')

INSERT INTO Customer (Id, Name, City)
VALUES  (2, 'Vero Strumica', 'Strumica')
```

READ operation

- Read all data in the table:

```
SELECT * FROM Customer
```

- Read only specific columns:

```
SELECT Id, Name, City
FROM Customer
```

- Read specific columns and Rows:

```
SELECT Id, Name, City
FROM Customer
WHERE City = 'Skopje'
```

UPDATE operation

- Update data

```
UPDATE TableName
SET Col1 = NewValue, Col2 = NewValue2,...
WHERE ColN = oldValue
```

- Example:

```
UPDATE Customer
SET Name = 'Vero Bitola', City = 'Bitola'
WHERE Name = 'Vero Skopje'
```

DELETE operation

- Delete data

```
DELETE
FROM TableName
WHERE ColN = oldValue
```

- Example:

```
DELETE
FROM Customer
WHERE Name = 'Vero Skopje'
```

Workshop – Creating tables

Creating tables by using T-SQL

Create new database: class_01

Create the following tables:

- Products
- *Users*

Try to insert data in the tables

Create table - Users

```sql
CREATE TABLE users (
    id INTEGER,
    username VARCHAR(50),
    email VARCHAR(100),
);
```

Create table - Products

```sql
CREATE TABLE products (
    id INTEGER,
    name VARCHAR(100),
    price DECIMAL(10,2),
);
```

Homework 1

QA Home - List of Tables

Student
- ID
- FirstName
- LastName
- DateOfBirth
- EnrolledDate
- Gender
- NationalIDNumber
- StudentCardNumber

Course
- ID
- Name
- Credit
- AcademicYear
- Semester

Teacher
- ID
- FirstName
- LastName
- DateOfBirth
- AcademicRank
- HireDate

Grade
- ID
- StudentID
- CourseID
- TeacherID
- Grade
- Comment
- CreatedDate

GradeDetails
- ID
- GradeID
- AchievementTypeID
- AchievementPoints
- AchievementMaxPoints
- AchievementDate

AchievementType
- ID
- Name
- Description
- ParticipationRate

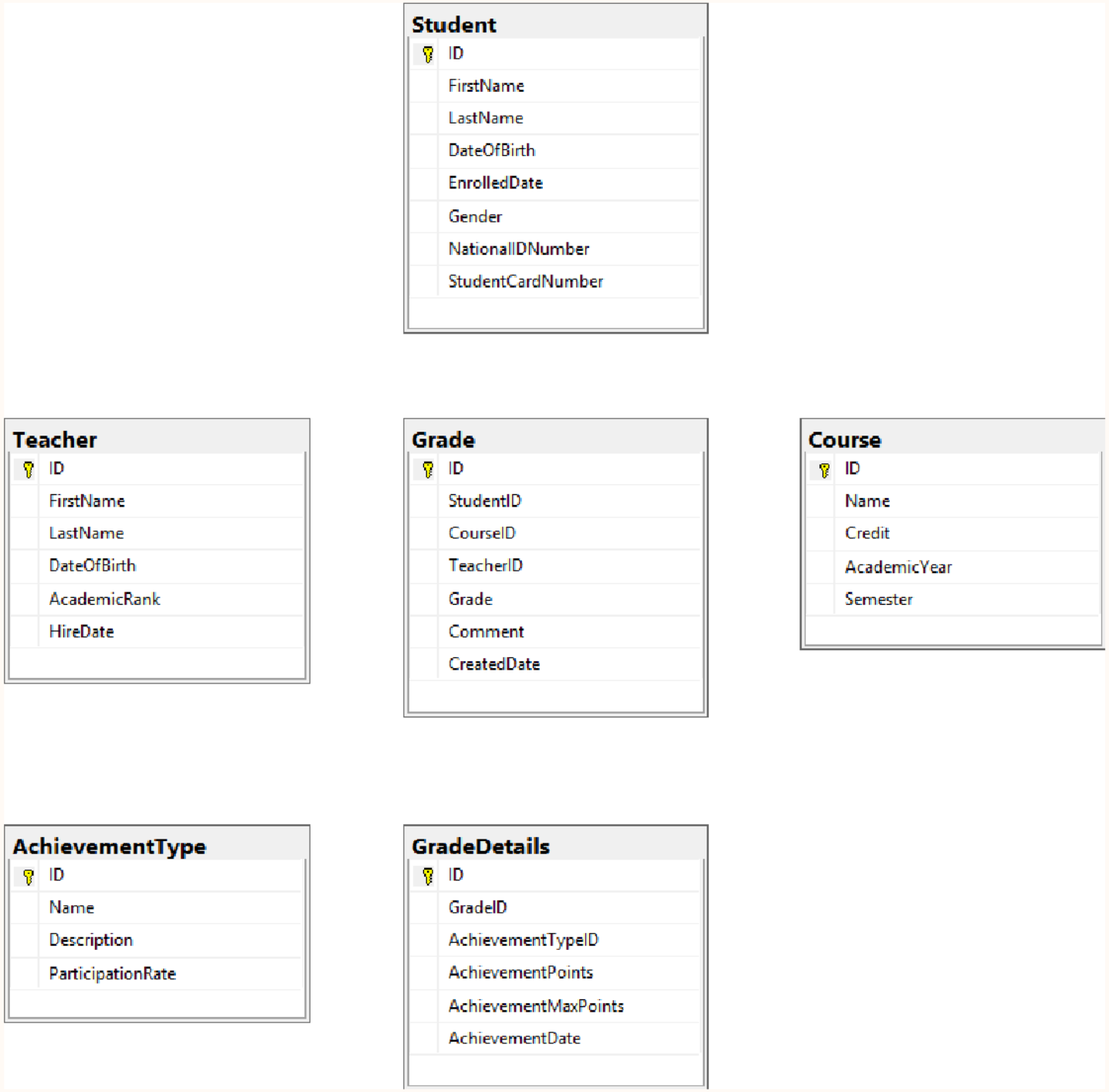
QA Home – Table example (Teacher)

Column name	Data type	Allow nulls
Id	Integer	
FirstName	varchar(20)	
LastName	varchar(30)	
DateOfBirth	date	
AcademicRank	varchar(20)	
HireDate	date	

QA Home – Table example (Grade)

Column name	Data type	Allow nulls
Id	Integer	
StudentId	Integer	
CourseId	Integer	
TeacherId	Integer	
Grade	smallint	
Comment	varchar(100)	
CreatedDate	date	

QA Home – Table Schema



Questions?

Trainer Name

Trainer mail

Assistant Name

Assistant mail

Qinshift Academy