

# Database Systems: A Practical Approach to Design, Implementation, and Management

Sixth Edition

Assignment Project Exam**Chapter 1**

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Learning Objectives (1 of 2)

1.1 Some common uses of database systems.

1.2 Characteristics of file-based systems.

Assignment Project Exam Help

1.3 Problems wit

1.4 Meaning of th <https://eduassistpro.github.io/>

1.5 Meaning of the term Database System  
(DBMS).

# Learning Objectives (2 of 2)

- 1.6 Typical functions of a DBMS.
- 1.7 Major components of the DBMS environment.  
[Assignment](#) [Project](#) [Exam](#) [Help](#)
- 1.8 Personnel involved in the DBMS environment.
- 1.9 History of the DBMS industry.  
<https://eduassistpro.github.io/>
- 1.10 Advantages and disadvantages of DBMSs.  
[Add WeChat](#) [edu\\_assist\\_pro](#)

# Examples of Database Applications

- Purchases from the supermarket
- Purchases using your credit card  
**Assignment Project Exam Help**
- Booking a holiday
- Using the local <https://eduassistpro.github.io/>
- Taking out insurance  
**Add WeChat edu\_assist\_pro**
- Renting a video
- Using the Internet
- Studying at university

# File-Based Systems

- Collection of application programs that perform services for the end users (e.g. reports).
- Each program defines and manages its own data.

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# File-Based Processing

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Sales Files

**PropertyForRent** (propertyNo, street, city, postcode, type, rooms, rent, and ownerNo)

**PrivateOwner** (ownerNo, fName, lName, address, telNo)

**Client** (clientNo, fName, lName, address, telNo, prefType, maxRent)

Contracts files

**Lease** (leaseNo, propertyNo, clientNo, rent, paymentMethod, deposit, paid, rentStart, rentFinish, duration)

**PropertyForRent** (propertyNo, street, city, postcode, rent)

**Client** (clientNo, fName, lName, address, telNo)

# Limitations of File-Based Approach (1 of 2)

- Separation and isolation of data
  - Each program maintains its own set of data.
  - Users of one program may be unaware of potentially useful data  
<https://eduassistpro.github.io/>
- Duplication of data
  - Same data is held by different programs.
  - Wasted space and potentially different values and/or different formats for the same item.

# Limitations of File-Based Approach (2 of 2)

- Data dependence
  - File structure is defined in the program code.
- Incompatible file formats
  - Programs at <https://eduassistpro.github.io/> and so cannot easily access each other's programs.
- Fixed Queries/Proliferation of programs
  - Programs are written to satisfy particular functions.
  - Any new requirement needs a new program.

# Database Approach (1 of 3)

- Arose because:
  - Definition of data was embedded in application programs, rather than being stored separately and independently
  - No control over the organization of data beyond that imposed by application programs.
- Result:
  - the database and Database Management System (DBMS).

# Database

- Shared collection of logically related data (and a description of this data), designed to meet the information needs of an organization
- System catalog to enable programs
- Logically related data comprising attributes, and relationships of an organization's information.

# Database Management System (DBMS) (1 of 2)

- A software system that enables users to define, create, maintain, and control access to the database.
- (Database) application program: a computer program that interacts with d  
(SQL statemen <https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Database Management System (DBMS) (2 of 2)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

**PropertyForRent** (propertyNo, street, city, postcode, type, rooms, rent, ownerNo)

**PrivateOwner** (ownerNo, fname, lName, address, telNo)

**Client** (clientNo, fName, lName, address, telNo, prefType, maxRent)

**Lease** (leaseNo, propertyNo, clientNo, paymentMethod, deposit, paid, rentStart, rentFinish)

# Database Approach (2 of 3)

- Data definition language (DDL).
  - Permits specification of data types, structures and any data constraints.
  - All specifica
- Data manipulation language (
  - General enquiry facility (q) of the data.

# Database Approach (3 of 3)

- Controlled access to database may include:
  - a security system
  - an integrity system
  - a concurrent system
  - a recovery system
  - a user-accessible catalog.

# Views

- Allows each user to have his or her own view of the database.
- A view is essentially some subset of the database.

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Views - Benefits

- Reduce complexity
- Provide a level of security  
**Assignment Project Exam Help**
- Provide a mechanism to change the appearance of the database  
**https://eduassistpro.github.io/**
- Present a consistent view of the structure of the database, even if the underlying database is changed  
**Add WeChat to edu\_assist\_pro**

# Components of DBMS Environment (1 of 3)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Components of DBMS Environment (2 of 3)

- Hardware
  - Can range from a PC to a network of computers.
- Software
  - DBMS, open <https://eduassistpro.github.io/> (if necessary) and also the application programs.  
**Add WeChat edu\_assist\_pro**
- Data
  - Used by the organization and a description of this data called the schema.

# Components of DBMS Environment (3 of 3)

- Procedures
  - Instructions and rules that should be applied to the design and use of the database and DBMS.

- People

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Roles in the Database Environment

- Data Administrator (DA)
- Database Administrator (DBA)  
[Assignment](#) [Project](#) [Exam](#) [Help](#)
- Database Desi  
    (a) Application Pro  
    (b) End Users (naive and sophisticated)
- Application Pro  
<https://eduassistpro.github.io/>  
[Add WeChat](#) [edu\\_assist\\_pro](#)
- End Users (naive and sophisticated)

# History of Database Systems

- First-generation
  - Hierarchical and Network
- Second generation
  - Relational <https://eduassistpro.github.io/>
- Third generation
  - Object-Relational
  - Object-Oriented

# Advantages of DBMSs (1 of 2)

- Control of data redundancy
- Data consistency  
**Assignment Project Exam Help**  
**https://eduassistpro.github.io/**  
**Add WeChat edu\_assist\_pro**
- More information about data
- Sharing of data
- Improved data integrity
- Improved security
- Enforcement of standards
- Economy of scale

# Advantages of DBMSs (2 of 2)

- Balance conflicting requirements
- Improved data accessibility and responsiveness  
[Assignment](#) [Project](#) [Exam](#) [Help](#)
- Increased prod
- Improved maint  
<https://eduassistpro.github.io/> dependence
- Increased concurrency  
[Add WeChat](#) [edu\\_assist\\_pro](#)
- Improved backup and recovery services

# Disadvantages of DBMSs

- Complexity
- Size  
Assignment Project Exam Help  
<https://eduassistpro.github.io/>
- Cost of DBMS
- Additional hard  
Add WeChat edu\_assist\_pro
- Cost of conversion
- Performance
- Higher impact of a failure

# Copyright

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Database Systems: A Practical Approach to Design, Implementation, and Management

Sixth Edition

Assignment Project Exam**Chapter 2**

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Learning Objectives (1 of 2)

# Learning Objectives (2 of 2)

- 2.7** Purpose/importance of conceptual modeling.
- 2.8** Typical functions and services a DBMS should provide.  
**Assignment Project Exam Help**
- 2.9** Function and catalog.
- 2.10** Software co <https://eduassistpro.github.io/>
- 2.11** Meaning of client–server and advantages of this type of architecture for a DBMS.  
**Add WeChat edu\_assist\_pro**
- 2.12** Function and uses of Transaction Processing Monitors.

# Objectives of Three-Level Architecture (1 of 2)

- All users should be able to access same data.
- A user's view is immune to changes made in other views.  
**Assignment Project Exam Help**
- Users should not store details of database storage. <https://eduassistpro.github.io/>

Add WeChat **edu\_assist\_pro**

# Objectives of Three-Level Architecture (2 of 2)

- DBA should be able to change database storage structures without affecting the users' views.
- Internal structure of database should be unaffected by changes to phy <https://eduassistpro.github.io/>
- DBA should be able to change physical structure of database without affecting all

# **ANSI-SPARC Three-Level Architecture (1 of 3)**

**Assignment Project Exam Help**

**<https://eduassistpro.github.io/>**

**Add WeChat edu\_assist\_pro**

# ANSI-SPARC Three-Level Architecture (2 of 3)

- External Level
  - Users' view of the database.
  - Describes the part of database that is relevant to a particular user.  
<https://eduassistpro.github.io/>
- Conceptual Level
  - Community view of the data
  - Describes what data is stored in database and relationships among the data.

# ANSI-SPARC Three-Level Architecture (3 of 3)

- Internal Level
  - Physical representation of the database on the computer
  - Describes how the database.

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Differences Between Three Levels of ANSI-SPARC Architecture

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Data Independence (1 of 2)

- Logical Data Independence
  - Refers to immunity of external schemas to changes in conceptual schema
  - Conceptual entities). addition/removal of
  - Should not require changes to all schema or rewrites of application programs.

# Data Independence (2 of 2)

- Physical Data Independence
  - Refers to immunity of conceptual schema to changes in the internal schema.
  - Internal schema organization (e.g., different file structures).
  - Should not require changes in external schemas.

# Data Independence and the ANSI-SPARC Three-Level Architecture

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Database Languages (1 of 2)

- Data Definition Language (DDL)
  - Allows the DBA or user to describe and name entities, attributes and relationships required for the application
  - plus any as <https://eduassistpro.github.io/> security constraints.

Add WeChat edu\_assist\_pro

# Database Languages (2 of 2)

- Data Manipulation Language (DML)
  - Provides basic data manipulation operations on data held in **Assignment Project Exam Help**
- Procedural DM <https://eduassistpro.github.io/>
  - allows user to tell system to manipulate data. **Add WeChat edu\_assist\_pro**
- Non-Procedural DML
  - allows user to state what data is needed rather than how it is to be retrieved.
- Fourth Generation Languages (4GLs)

# Data Model (1 of 2)

- Integrated collection of concepts for describing data, relationships between data, and constraints on the data in an organization
- Data Model consists of:
  - a structural part;
  - a manipulative part;
  - possibly a set of integrity rules.

# Data Model (2 of 2)

- Purpose
  - To represent data in an understandable way.
- Categories of data models include:
  - Object-base
  - Record-based
  - Physical.

# Data Models

- Object-Based Data Models
  - Entity-Relationship
  - Semantic [Assignment Project Exam Help](#)
  - Functional <https://eduassistpro.github.io/>
  - Object-Orie  
[Add WeChat edu\\_assist\\_pro](#)
- Record-Based Data Models
  - Relational Data Model
  - Network Data Model
  - Hierarchical Data Model.
- Physical Data Models

# Relational Data Model

## Branch

branchNo	street	City	postCode
B005	22 Deer Rd	London	SW1 4EH
B007	10 Argyll St	Aberdeen	AB2 3SU
B003	163		9QX
B004	32		1HZ
B002	56		0 6EU

## Staff

Add WeChat edu\_assist\_pro

staffNo	fName	IName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

# Network Data Model

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Hierarchical Data Model

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Conceptual Modeling

- Conceptual schema is the core of a system supporting all user views.
- Should be complete and accurate representation of an organization's data.  
<https://eduassistpro.github.io/>
- Conceptual modeling is proceeding a model of information use that is independent of implementation details.
- Result is a conceptual data model.

# Functions of a DBMS (1 of 2)

- Data Storage, Retrieval, and Update.
- A User-Accessible Catalog.  
[Assignment](#) [Project](#) [Exam](#) [Help](#)
- Transaction Su
- Concurrency C <https://eduassistpro.github.io/>
- Recovery Services.  
[Add WeChat edu\\_assist\\_pro](#)

# Functions of a DBMS (2 of 2)

- Authorization Services.
- Support for Data Communication.  
[Assignment](#) [Project](#) [Exam](#) [Help](#)
- Integrity Services
- Services to Pros  
<https://eduassistpro.github.io/>  
e.
- Utility Services.  
[Add WeChat edu\\_assist\\_pro](#)

# System Catalog

- Repository of information (metadata) describing the data in the database.
- One of the fundamental components of DBMS.
- Typically stores <https://eduassistpro.github.io/>
  - names, types, and sizes of data items;
  - constraints on the data;
  - names of authorized users;
  - data items accessible by a user and the type of access;
  - usage statistics.

# Components of a DBMS

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Components of Database Manager

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Multi-User DBMS Architectures

- Teleprocessing
- File-server
- Client-server

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Teleprocessing

- Traditional architecture.
- Single mainframe with a number of terminals attached.  
**Assignment Project Exam Help**
- Trend is now to

<https://eduassistpro.github.io/>

Add WeChat **edu\_assist\_pro**

# File-Server

- File-server is connected to several workstations across a network.
- Database resides on file-server.
- DBMS and app <https://eduassistpro.github.io/>.
- Disadvantages include:
  - Significant network traffic.
  - Copy of DBMS on each workstation.
  - Concurrency, recovery and integrity control more complex.

# File-Server Architecture

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Traditional Two-Tier Client-Server (1 of 3)

- Client (tier 1) manages user interface and runs applications.
- Server (tier 2) holds database and DBMS.
- Advantages include:
  - wider access to existing data;
  - increased performance;
  - possible reduction in hardware costs;
  - reduction in communication costs;
  - increased consistency.

# Traditional Two-Tier Client-Server (2 of 3)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Traditional Two-Tier Client-Server (3 of 3)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Three-Tier Client-Server (1 of 3)

- Client side presented two problems preventing true scalability:
  - ‘Fat’ client, requiring considerable resources on client’s com
  - Significant c <https://eduassistpro.github.io/> overhead.
- By 1995, three layers proposed potentially running on a different platform.
  - Assignment Project Exam Help
  - Add WeChat edu\_assist\_pro

# Three-Tier Client-Server (2 of 3)

- Advantages:
  - ‘Thin’ client, requiring less expensive hardware.
  - Application maintenance centralized.
  - Easier to maintain without affecting others.
  - Separating business logic from presentation functions makes it easier to implement load balancing.
  - Maps quite naturally to Web environment.

# Three-Tier Client-Server (3 of 3)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Transaction Processing Monitors

- Program that controls data transfer between clients and servers in order to provide a consistent environment, particularly for Online Transaction Processing (OLTP).

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# TPM as Middle Tier of 3-Tier Client-Server

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Copyright

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Database Systems: A Practical Approach to Design, Implementation, and Management

Sixth Edition

Assignment Project Exam**Chapter 3**

<https://eduassistpro.github.io/>

Database Architectures and  
Transparencies

Add WeChat edu\_assist\_pro

# Learning Objectives (1 of 2)

**3.1** The meaning of the client–server architecture and the advantages of this type of architecture for a DBMS.

**3.2** The difference between two-tier, three-tier and n-tier client–server arc

<https://eduassistpro.github.io/>

**3.3** About cloud computing and service (DaaS) and database as a service (DB)

Add WeChat [edu\\_assist\\_pro](http://edu_assist_pro)

**3.4** Software components of a DBMS.

# Learning Objectives (2 of 2)

**3.5** The purpose of a Web service and the technological standards used to develop a Web service.

**3.6** The meaning of service-oriented architecture (SOA).

**3.7** The difference between集中式 and distributed processing.

Add WeChat edu\_assist\_pro

**3.8** The architecture of a data w

**3.9** About cloud computing and cloud databases.

**3.10** The software components of a DBMS.

# Multi-User DBMS Architectures

- The common architectures that are used to implement multi-user database management systems:
  - Teleprocessing Assignment Project Exam Help
  - File-Server
  - Client-Serv

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

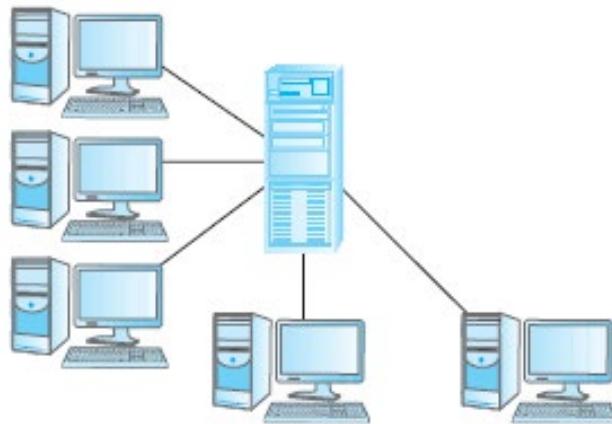
# File-Server

- File-server is connected to several workstations across a network.
- Database resides on file-server.
- DBMS and app <https://eduassistpro.github.io/>.
- Disadvantages include:
  - Significant network traffic.
  - Copy of DBMS on each workstation.
  - Concurrency, recovery and integrity control more complex.

# Teleprocessing

- One computer with a single CPU and a number of terminals.
- Processing performed within the same physical computer. User “dumb”, incapable of functioning o <https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro



# File-Server Architecture

- In a file-server environment, the processing is distributed about the network, typically a local area network (LAN).

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Traditional Two-Tier Client-Server (1 of 3)

- Client (tier 1) manages user interface and runs applications.
- Server (tier 2) holds database and DBMS.
- Advantages include:
  - wider access to existing data;
  - increased performance;
  - possible reduction in hardware costs;
  - reduction in communication costs;
  - increased consistency.

# Traditional Two-Tier Client-Server (2 of 3)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Alternative Client-Server Topologies

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Traditional Two-Tier Client-Server (3 of 3)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Summary of Client-Server Functions

Client	Server
Manages the user interface	Accepts and processes database requests from clients
Accepts and checks syntax of user input	Checks authorization
Processes application logic	Ensures constraints not violated
Generates database request to server	Performs processing and transmits response
Passes response back to user	Maintains database integrity and provides concurrent database access

# Three-Tier Client-Server (1 of 3)

- The need for enterprise scalability challenged the traditional two-tier client–server model.
- Client side presented two problems preventing true scalability:
  - ‘Fat’ client, running on client’s computer to run off <https://eduassistpro.github.io/> sources on Add WeChat to edu\_assist\_pro
  - Significant client side administration overhead.
- By 1995, three layers proposed, each potentially running on a different platform.

# Three-Tier Client-Server (2 of 3)

- Advantages:
  - ‘Thin’ client, requiring less expensive hardware.
  - Application maintenance centralized.
  - Easier to maintain without affecting others.
  - Separating business logic from presentation functions makes it easier to implement load balancing.
  - Maps quite naturally to Web environment.

# Three-Tier Client-Server (3 of 3)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# n-Tier Client-Server (e.g. 4-Tier)

- The three-tier architecture can be expanded to **n** tiers, with additional tiers providing more and scalability. <https://eduassistpro.github.io/>

**Add WeChat edu\_assist\_pro**

- Applications servers host API to expose business logic and business processes for use by other applications.

# Middleware

- Middleware is a generic term used to describe software that mediates with other software and allows for communication between disparate applications in a heterogeneous environment.
- The need for middleware in distributed systems becomes too complex to manage effectively without a common interface.  
<https://eduassistpro.github.io/>  
Add WeChat edu\_assist\_pro

# Cloud Computing (1 of 2)

- The National Institute of Standards and Technology (NIST) provided a definition.
- Defined as "A model for enabling ubiquitous, convenient, on-demand net <https://eduassistpro.github.io/> configurable co pool of networks, servers, storage, applications, and so provisioned and released with Add WeChat [edu\\_assist\\_pro](#) han be rapidly or service provider interaction."

# Transaction Processing Monitors

- TP monitor is a program that controls data transfer between clients and servers in order to provide a consistent environment, particularly for online transaction processing (OLP)

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Transaction Processing Monitor as Middle Tier of 3-Tier Client-Server

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Web Services and Service-Oriented Architectures (1 of 3)

- Web service is a software system designed to support interoperable machine-to-web service machine interaction over a network.
- Web services send and receive data, and processes through a program, a, and processes a network.
- Developers can add the Web page (or an executable program) to offer specific functionality to users.

# Web Services and Service-Oriented Architectures (2 of 3)

- Web services approach uses accepted technologies and standards, such as:
  - XML (eXtensible Markup Language)
  - SOAP (Simple Object Access Protocol) is a communication protocol based on XML. It is used to exchange structured information over the Internet. It defines a message format based on XML. It is language-independent.
  - WSDL (Web Services Description Language) is a protocol, again based on XML, used to describe and locate a Web service.

# Web Services and Service-Oriented Architectures (3 of 3)

- UDDI (Universal Discovery, Description, and Integration) protocol is a platform independent, XML-based registry for businesses to list themselves on the Internet.

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Service-Oriented Architectures (SOA)

- A business-centric software architecture for building applications that implement business processes as sets of services published at a granularity relevant to the service consumer. These services are defined, published, and discovered using a single implementation based form of interface.

# Distributed DBMSs (1 of 2)

- A distributed database is a logically interrelated collection of shared data (and a description of this data), physically distributed over a computer network
- A distributed D the manageme tem that permits base and makes the distribution transparent to Add WeChat edu\_assist\_pro

# Distributed DBMSs (2 of 2)

- A DDBMS consists of a single logical database split into a number **of fragments**.
- Each fragment is stored on one or more computers (**replicas**) under <https://eduassistpro.github.io/> the DBMS, with the computers con
- Each site is capable of indepe Add WeChat [edu\\_assist\\_pro](https://edu_assist_pro) cessing user requests that require access to local data (that is, each site has some degree of local autonomy) and is also capable of processing data stored on other computers in the network.

# Data Warehousing

- A data warehouse was deemed the solution to meet the requirements of a system capable of supporting decision making, receiving data from multiple operational data sources.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Cloud Computing (2 of 2)

- The National Institute of Standards and Technology (NIST) provided a definition.
- Defined as "A model for enabling ubiquitous, convenient, on-demand net <https://eduassistpro.github.io/> configurable co pool of networks, servers, storage, applications, and so on that can be rapidly provisioned and released with or service provider interaction."

# Cloud Computing – Key Characteristics (1 of 3)

- **On-demand self-service**
  - Consumers can obtain, configure and deploy cloud services without help from provider.
- **Broad network**<https://eduassistpro.github.io/>
  - Accessible from anywhere using standardized platform (e.g. desktop, mobile devices).

# Cloud Computing – Key Characteristics (2 of 3)

- **Resource pooling**
  - Provider's computing resources are pooled to serve multiple consumers with different physical and virtual resources and reassigned according to consumer demand. Examples of resources include storage, processing, memory, and network bandwidth.

# Cloud Computing – Key Characteristics (3 of 3)

- **Rapid elasticity**
  - Provider's capacity caters for customer's spikes in demand and reduces risk of outages and service interruption
    - able to scale rapidly based on https://eduassistpro.github.io/
- **Measured service**
  - Provider uses a metering capability to measure usage of service (e.g. storage, processing, bandwidth, and active user accounts).

# Cloud Computing – Service Models (1 of 3)

- **Software as a Service (SaaS):**
  - Software and data hosted on cloud. Accessed through client interface (e.g. web browser). Consumer application <https://eduassistpro.github.io/>
  - Examples include sales management applications, NetSuite's integrated business management software, Google's Gmail and Cornerstone OnDemand.

# Cloud Computing – Service Models (2 of 3)

- **Platform as a Service (PaaS)**

- Allows creation of web applications without buying/maintaining the software and underlying infrastructure including new <https://eduassistpro.github.io/> storage, while customer controls deployment and possibly configuration.
- Examples include Salesforce.com's Force.com, Google's App Engine, and Microsoft's Azure.

# Cloud Computing – Service Models (3 of 3)

- **Infrastructure as a Service (IaaS)**
  - Provider's offer servers, storage, network and operating systems – Typically virtualization environments as on-demand services, in a pay-as-you-go model according to usage.
  - A popular use of IaaS is in enterprise sites. Examples Amazon's Elastic Compute Cloud (EC2), Rackspace and GoGrid.

# Cloud Computing – Comparison of Services Models

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Benefits of Cloud Computing (1 of 2)

- **Cost-Reduction:** Avoid up-front capital expenditure.
- **Scalability/Agility:** Organisations set up resources on an as-needs basis.  
*Assignment Project Exam Help*
- **Improved Sec** <https://eduassistpro.github.io/> ote expertise & resources to security; not afford able by customer.  
*Add WeChat edu\_assist\_pro*
- **Improved Reliability:** Provide resources on reliability of systems; not affordable by customer.
- **Access to new technologies:** Through use of provider's systems, customers may access latest technology.

# Benefits of Cloud Computing (2 of 2)

- **Faster development:** Provider's platforms can provide many of the core services to accelerate development cycle.  
[Assignment Project Exam Help](https://eduassistpro.github.io/)
- **Large scale pr** : Providers have the resources t <https://eduassistpro.github.io/>
- **More flexible working practi** Add WeChat [edu\\_assist\\_pro](https://edu_assist_pro) an access files using mobile devices.
- **Increased competitiveness:** Allows organizations to focus on their core competencies rather than their IT infrastructures.

# Risks of Cloud Computing

- **Network Dependency:** Power outages, bandwidth issues and service interruptions.
- **System Dependency:** Customer's dependency on availability and reliability of provider.
- **Cloud Provider D** <https://eduassistpro.github.io/> became insolvent or acquired by competitor, resulting in sudden termination.  
**Add WeChat edu\_assist\_pro**
- **Lack of control:** Customers unable to control technical or organisational measures to safeguard the data. May result in reduced availability, integrity, confidentiality, intervenability and isolation.
- **Lack of information on processing transparency**

# Cloud-Based Database Solutions (1 of 6)

- As a type of Software as a Service (SaaS), cloud-based database solutions fall into two basic categories:
  - Data as ~~Assignment Project Exam Help~~
  - Database a <https://eduassistpro.github.io/>
- Key difference between the two mainly how the data is managed.  
[Add WeChat edu\\_assist\\_pro](#)

# Cloud-Based Database Solutions (2 of 6)

- DBaaS
  - Offers full database functionality to application developers
  - Provides a continuous database to support optimized scalability, availability, multi-tenancy (that is, serving multiple organizations), and effective resource allocation in the cloud, thereby sparing the developer from ongoing database administration tasks.

# Cloud-Based Database Solutions (3 of 6)

- DaaS:
  - Services enables data definition in the cloud and subsequently Project Exam Help
  - Does not implement interfaces (e.g. SQL) but instead common APIs. <https://eduassistpro.github.io/>
  - Enables organization with edu\_assist\_pro to offer access to others. Examples Urban Mapping (geography data service), Xignite (financial data service) and Hoovers (business data service.)

# Cloud-Based Database Solutions (4 of 6)

- Multi-tenant cloud database-shared server, separate database server process architecture.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Cloud-Based Database Solutions (5 of 6)

- Multi-tenant cloud database-shared DBMS server, separate databases.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Cloud-Based Database Solutions (6 of 6)

- Multi-tenant cloud database—shared database, separate schema architecture.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Components of a DBMS (1 of 5)

- A DBMS is partitioned into several software components (or **modules**), each of which is assigned a specific operation. As stated previously, some of the functions of the DBMS are managing operating system. <https://eduassistpro.github.io/>
- The DBMS interfaces with other components, such as user queries and access management techniques for storing and retrieving data records).

# Components of a DBMS (2 of 5)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Components of a DBMS (3 of 5)

- **Query processor** is a major DBMS component that transforms queries into a series of low-level instructions directed to the database manager.  
<https://eduassistpro.github.io/>
- **Database manager** application processes user-submitted requests. The DM examines the external and conceptual schema to determine what conceptual records are required to fulfill the request. The DM then places a call to the file manager to perform the request.

# Components of a DBMS (4 of 5)

- **File manager** manipulates the underlying storage files and manages the allocation of storage space on disk. It establishes and maintains the list of structures and indexes defined
- **DML preprocessor** elements embedded in an application program into function calls in the host language. The DML processor must interact with the query processor to generate the appropriate code.

# Components of a DBMS (5 of 5)

- **DDL compiler** converts DDL statements into a set of tables containing metadata. These tables are then stored in the system catalog while control information is stored in data file header.
- **Catalog manager** maintains the system catalog. The system catalog is accessed by most DBMS components.

# Components of Database Manager (DM)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Components of the Database Manager (1 of 3)

- **Authorization control** to confirm whether the user has the necessary permission to carry out the required operation. [Assignment](#) [Project](#) [Exam](#) [Help](#)
- **Command processor** <https://eduassistpro.github.io/> of user authority, control is passed to the command processor.
- **Integrity checker** ensures that the operation satisfies all necessary integrity constraints (e.g. key constraints) for an operation that changes the database.

# Components of the Database Manager (2 of 3)

- **Query optimizer** determines an optimal strategy for the query execution.
- **Transaction manager** performs the required processing of operations through actions.  
<https://eduassistpro.github.io/>
- **Scheduler** ensures that concurrent transactions on the database proceed without conflicting one another. It controls the relative order in which transaction operations are executed.

# Components of the Database Manager (3 of 3)

- **Recovery manager** ensures that the database remains in a consistent state in the presence of failures. It is responsible for transaction commit and abort.
- **Buffer manager** between main memory and storage, such as disk and tape. The recovery manager and the buffer manager also known as (aka) the **data manager**. The buffer manager aka the **cache manager**.

<https://eduassistpro.github.io/>

# Copyright

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Database Systems: A Practical Approach to Design, Implementation, and Management

Sixth Edition

Assignment Project Exam**Chapter 4**

<https://eduassistpro.github.io/> Relational Model

Add WeChat edu\_assist\_pro

# Learning Objectives

- 4.1 Terminology of relational model.**

**4.2 How tables are used to represent data.**

**Assignment Project Exam Help**

**4.3 Connection b** lations and  
relations in the re<https://eduassistpro.github.io/>

**4.4 Properties of database relations** Add WeChat [edu\\_assist\\_pro](https://edu_assist_pro)

**4.5 How to identify CK, PK, and FKs.**

**4.6 Meaning of entity integrity and referential integrity.**

**4.7 Purpose and advantages of views.**

# Relational Model Terminology (1 of 2)

- A relation is a table with columns and rows.
  - Only applies to logical structure of the database, not the physical structure
- Attribute is a name
- Domain is the set of allowable values for one or more attributes.

# Relational Model Terminology (2 of 2)

- Tuple is a row of a relation.
- Degree is the number of attributes in a relation.  
[Assignment](#) [Project](#) [Exam](#) [Help](#)
- Cardinality is the number of tuples in a relation.
- Relational Data is organized into relations with distinct relation names.  
<https://eduassistpro.github.io/>  
[Add](#) [WeChat](#) [edu\\_assist\\_pro](#)

# Instances of Branch and Staff Relations

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Examples of Attribute Domains

Attribute	Domain Name	Meaning	Domain Definition
branchNo	BranchNumbers	The set of all possible branch numbers	character: size 4, range B001–B999
street	StreetNames	The set of all street names in	character: size 25
city	CityNames	<a href="https://eduassistpro.github.io/">https://eduassistpro.github.io/</a>	character: size 15
postcode	Postcodes	The set of all post	character: size 1, value M or F
sex	Sex	The sex of a person	character: size 1, value M or F
DOB	DatesOfBirth	Possible value of staff birth dates	date, range from 1-Jan-20, format dd-mmm-yy
salary	Salaries	Possible values of staff salaries	monetary: 7 digits, range 6000.00–40000.00

# Alternative Terminology for Relational Model

Formal terms	Alternative 1	Alternative 2
Relation	Table	File
Tuple	Row	Record
Attribute	<a href="https://eduassistpro.github.io/">https://eduassistpro.github.io/</a>	

Add WeChat edu\_assist\_pro

# Mathematical Definition of Relation (1 of 4)

- Consider two sets,  $D_1$  &  $D_2$ , where  $D_1 = \{2, 4\}$  and  $D_2 = \{1, 3, 5\}$ .  
**Assignment Project Exam Help**  
**Add WeChat edu\_assist\_pro**
- Cartesian product,  $D \times D$ , is set of all ordered pairs, where first element is member of  $D_1$  and second element is member of  $D_2$ .  
$$D_1 \times D_2 = \{(2, 1), (2, 3), (2, 5), (4, 1), (4, 3), (4, 5)\}$$
- Alternative way is to find all combinations of elements with first from  $D_1$  and second from  $D_2$ .

# Mathematical Definition of Relation (2 of 4)

- Any subset of Cartesian product is a relation; e.g.

$$R = \{(2,1), (4,1)\}$$

- May specify which pairs are in relation using some condition for sel
  - second ele

$$R = \{(x,y) \mid x \in D_1, y \in D_2, \text{ and } y = 1\}$$

- first element is always twice the second:

$$S = \{(x,y) \mid x \in D_1, y \in D_2, \text{ and } x = 2y\}$$

## Mathematical Definition of Relation (3 of 4)

- Consider three sets  $D_1, D_2, D_3$  with Cartesian Product  $D_1 \times D_2 \times D_3$ ; e.g.

Assignment Project Exam Help

$$D_1 = \{1, 3\}$$

$$D_1 \times D_2 \times D_3 = \{(1, 2, 5),$$

$$(1, 4, 6), (3, 2, 5), (3, 2, 6), (3, 4, 6)\}$$

- Any subset of these ordered triples is a relation.

# Mathematical Definition of Relation (4 of 4)

- Cartesian product of  $n$  sets  $(D_1, D_2, \dots, D_n)$  is:

$D_1 \times D_2 \times \dots \times D_n = \{(d_1, d_2, \dots, d_n) | d_1 \in D_1, d_2 \in D_2, \dots, d_n \in D_n\}$

Assignment Project Exam Help

usually written

$$\prod_{i=1}^n D_i$$

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- Any set of  $n$ -tuples from this Cartesian product is a relation on the  $n$  sets.

# Database Relations

- Relation schema
  - Named relation defined by a set of attribute and domain name pairs
- Relational data
  - Set of relation schemas, each having a distinct name.

# Properties of Relations (1 of 2)

- Relation name is distinct from all other relation names in relational schema.
- Each cell of relation contains exactly one atomic (single) value.  
<https://eduassistpro.github.io/>
- Each attribute has a distinct n  
[Add WeChat edu\\_assist\\_pro](#)
- Values of an attribute are all fr  
e domain.

# Properties of Relations (2 of 2)

- Each tuple is distinct; there are no duplicate tuples.
- Order of attributes has no significance.
- Order of tuples

Assignment Project Exam Help

oretically.

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Relational Keys (1 of 2)

- Superkey
  - An attribute, or set of attributes, that uniquely identifies a tuple within a relation.
- Candidate Key <https://eduassistpro.github.io/>
  - Superkey (K) such that no set is a superkey within the relation.
  - In each tuple of R, values of K uniquely identify that tuple (uniqueness).
  - No proper subset of K has the uniqueness property (irreducibility).

# Relational Keys (2 of 2)

- Primary Key
  - Candidate key selected to identify tuples uniquely within relation.
- Alternate Keys <https://eduassistpro.github.io/>
  - Candidate keys that are not chosen to be primary key.
- Foreign Key
  - Attribute, or set of attributes, within one relation that matches candidate key of some (possibly same) relation.

# Integrity Constraints (1 of 3)

- Null
  - Represents value for an attribute that is currently unknown or not applicable for tuple.
  - Deals with invalid data.
  - Represents and is not the same as zero or spaces.

# Integrity Constraints (2 of 3)

- Entity Integrity
  - In a base relation, no attribute of a primary key can be null. [Assignment](#) [Project](#) [Exam](#) [Help](#)
- Referential Integrity <https://eduassistpro.github.io/>
  - If foreign key exists in a relation, its value must match a candidate key value of some tuple in its home relation or foreign key value must be wholly null.

# Integrity Constraints (3 of 3)

- General Constraints
  - Additional rules specified by users or database administrators that define or constrain some aspect of the enterprise

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Views (1 of 2)

- Base Relation
  - Named relation corresponding to an entity in conceptual schema whose tuples are physically stored in database
- View
  - Dynamic result of one or more operations operating on base relations to produce another relation.

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

## Views (2 of 2)

- A virtual relation that does not necessarily actually exist in the database but is produced upon request, at time of request. [Assignment Project Exam Help](https://eduassistpro.github.io/)
- Contents of a view are based on one or more base relations. <https://eduassistpro.github.io/>
- Views are dynamic, meaning that changes made to base relations that affect view attributes are immediately reflected in the view. [Add WeChat edu\\_assist\\_pro](https://eduassistpro.github.io/)

# Purpose of Views

- Provides powerful and flexible security mechanism by hiding parts of database from certain users.
- Permits users to access data in a customized way, so that same data can be presented to users in different ways, at same time.  
<https://eduassistpro.github.io/>
- Can simplify complex operations.

# Updating Views (1 of 3)

- All updates to a base relation should be immediately reflected in all views that reference that base relation.
- If view is updated, underlying base relation should reflect change.

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

## Updating Views (2 of 3)

- There are restrictions on types of modifications that can be made through views:
  - Updates are allowed if query involves a single base relation and key of base relation.
  - Updates are allowed involving multiple base relations. Add WeChat `edu_assist_pro`
  - Updates are not allowed involving aggregation or grouping operations.

# Updating Views (3 of 3)

- Classes of views are defined as:
  - theoretically not updateable;
  - theoretically updateable;
  - partially upd

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Copyright

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Database Systems: A Practical Approach to Design, Implementation, and Management

Sixth Edition

Assignment Project Exam**Chapter 5**

<https://eduassistpro.github.io/>

Final Algebra and  
Final Calculus

Add WeChat **edu\_assist\_pro**

# Learning Objectives

5.1 Meaning of the term relational completeness.

5.2 How to form queries in relational algebra.

Assignment Project Exam Help

5.3 How to form I calculus.

5.4 How to form <https://eduassistpro.github.io/> nal calculus.

5.5 Categories of relational DM Add WeChat edu\_assist\_pro

# Introduction

- Relational algebra and relational calculus are formal languages associated with the relational model.
- Informally, relational algebra is a (high-level) procedural language and r <https://eduassistpro.github.io/> -procedural language.
- However, formally both are equivalent to one another.
- A language that produces a relation that can be derived using relational calculus is **relationally complete**.

# Relational Algebra (1 of 2)

- Relational algebra operations work on one or more relations to define another relation without changing the original relations.
- Both operands of one operation can be output from another operation.  
<https://eduassistpro.github.io/>
- Allows expressions to be nested in arithmetic.  
This property is called **closure**.

# Relational Algebra (2 of 2)

- Five basic operations in relational algebra: Selection, Projection, Cartesian product, Union, and Set Difference.
- These perform most of the data retrieval operations needed.  
<https://eduassistpro.github.io/>
- Also have Join, Intersection, a  
which can be expressed in ter  
[Add WeChat edu\\_assist\\_pro](#) operations,

# Relational Algebra Operations (1 of 2)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Relational Algebra Operations (2 of 2)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Selection (or Restriction)

- $\sigma_{\text{predicate}}(R)$ 
  - Works on a single relation R and defines a relation that contains only those tuples (rows) of R that satisfy the specific

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Example - Selection (or Restriction)

- List all staff with a salary greater than £10,000.

$$\sigma_{\text{salary} > 10000}(\text{Staff})$$

Assignment Project Exam Help

staffNo	fName	l	https://eduassistpro.github.io/	salary	branchNo
SL21	John	White	Manager	-45	30000
SG37	Ann	Beech	Assistant	-60	12000
SG14	David	Ford	Supervisor	M	24-Mar-58
SG5	Susan	Brand	Manager	F	3-Jun-40

# Projection

- $\Pi_{\text{col1}, \dots, \text{coln}}(R)$ 
  - Works on a single relation R and defines a relation that contains a vertical subset of R, extracting the values of specific columns while eliminating duplicates.

Add WeChat edu\_assist\_pro

# Example - Projection

- Produce a list of salaries for all staff, showing only staffNo, fName, lName, and salary details.

$\prod_{\text{staffNo}, f}$  Assignment Project Exam Help  
(Staff)

<https://eduassistpro.github.io/>

staffNo	fName	lName	salary
SL21	John	Whit	0
SG37	Ann	Beech	12000
SG14	David	Ford	18000
SA9	Mary	Howe	9000
SG5	Susan	Brand	24000
SL41	Julie	Lee	9000

# Union

- $R \cup S$ 
  - Union of two relations R and S defines a relation that contains all the tuples of R, or S, or both R and S, duplicate tu
  - R and S mu <https://eduassistpro.github.io/>
- If R and S have  $I$  and  $J$  tuples, respectively, union is obtained by concatenating them into one relation with a maximum of  $(I + J)$  tuples.

# Example - Union

- List all cities where there is either a branch office or a property for rent.

$\Pi_{\text{city}} (\text{Branch}) \cup \Pi_{\text{city}} (\text{PropertyForRent})$

<https://eduassistpro.github.io/>

city	Add WeChat edu_assist_pro
London	
Aberdeen	
Glasgow	
Bristol	

# Set Difference

- $R - S$ 
  - Defines a relation consisting of the tuples that are in relation  $R$ , but not in  $S$ .
  - $R$  and  $S$  mu <https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Example - Set Difference

- List all cities where there is a branch office but no properties for rent.

$\Pi_{\text{city}} (\text{Branch}) - \Pi_{\text{city}} (\text{PropertyForRent})$

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

city
Bristol

# Intersection

- $R \cap S$ 
  - Defines a relation consisting of the set of all tuples that are in both R and S.
  - R and S must have the same arity.
- Expressed using:
$$R \cap S = R - (R - S)$$

# Example - Intersection

- List all cities where there is both a branch office and at least one property for rent.

$\Pi_{\text{city}} (\text{Branch}) \cap \Pi_{\text{city}} (\text{PropertyForRent})$

<https://eduassistpro.github.io/>

city
Aberdeen
London
Glasgow

# Cartesian Product

- $R \times S$ 
  - Defines a relation that is the concatenation of every tuple of ~~Assignment Project Exam Help~~ R with every tuple of relation S.

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Example - Cartesian Product

- List the names and comments of all clients who have viewed a property for rent.

$$(\Pi_{\text{clientNo}, \text{fName}, \text{lName}}(\text{Client}) \times (\Pi_{\text{clientNo}, \text{propertyNo}, \text{comment}}(\text{Viewing})))$$

client.clientNo	fName	lName	propertyNo	comment
CR76	John		PA14	too small
CR76	John	Kay	PG4	too remote
CR76	John	Kay	PG4	
CR76	John	Kay	CR62	no dining room
CR76	John	Kay	CR56	
CR56	Aline	Stewart	CR56	too small
CR56	Aline	Stewart	CR76	too remote
CR56	Aline	Stewart	CR56	
CR56	Aline	Stewart	CR62	no dining room
CR56	Aline	Stewart	CR56	

## Example - Cartesian Product (2 of 2)

client.clientNo	fName	IName	Viewing.clientNo	propertyNo	comment
CR74	Mike	Ritchie	CR56	PA14	too small
CR74	Mike	Ritchie	CR76	PC4	too remote
CR74	Mike			PG4	
CR74	Mike			FA14	no dining room
CR74	Mike	Ritchie	CR56	PG36	
CR62	Mary	Tregear	CR56	14	too small
CR62	Mary	Tregear	CR76	4	too remote
CR62	Mary	Tregear	CR56	PG4	
CR62	Mary	Tregear	CR62	PA14	no dining room
CR62	Mary	Tregear	CR56	PG36	

# Example - Cartesian Product and Selection

- Use selection operation to extract those tuples where Client.clientNo = Viewing.clientNo.

$\sigma_{\text{Client.clientNo} = \text{Viewing.ClientNo}}(\Pi_{\text{clientNo}, \text{fName}, \text{IName}}(\text{Client})) \times (\Pi_{\text{clientNo, property}} \text{Assignment Project Exam Help} (\text{Client}))$

<https://eduassistpro.github.io/>

client.clientNo	fName	IName	Viewi	pertyNo	Comment
CR76	John	Kay	CR7	4	too remote
CR56	Aline	Stewart	CR56	PA14	too small
CR56	Aline	Stewart	CR56	PG4	
CR56	Aline	Stewart	CR56	PG36	
CR62	Mary	Tregear	CR62	PA14	no dining room

- Cartesian product and Selection can be reduced to a single operation called a **Join**.

# Join Operations (1 of 2)

- Join is a derivative of Cartesian product.
- Equivalent to performing a Selection, using join predicate as selection formula, over Cartesian product of the two operand relations.  
<https://eduassistpro.github.io/>
- One of the most difficult operations to implement efficiently in an RDBMS and one of the reasons why RDBMSs have intrinsic performance problems.

# Join Operations (2 of 2)

- Various forms of join operation
  - Theta join
  - Equijoin (a particular type of Theta join)
  - Natural join <https://eduassistpro.github.io/>
  - Outer join
  - Semijoin [Add WeChat edu\\_assist\\_pro](#)

# Theta Join ( $\theta$ -Join) (1 of 2)

- $R \bowtie_F S$ 
  - Defines a relation that contains tuples satisfying the predicate  $F$  from the Cartesian product of  $R$  and  $S$ .
  - The predicate  $F$  is  $S.b_i \theta R.b_j$ , where  $\theta$  may be one of  $\{<, \leq, >, \geq, =, \neq\}$ .

<https://eduassistpro.github.io/>  
Add WeChat edu\_assist\_pro

## Theta Join ( $\theta$ -Join) (2 of 2)

- Can rewrite Theta join using basic Selection and Cartesian product operations.

Assignment Project Exam Help

- Degree of a Theta join is the sum of the operand relations R and S. If predicate equality (=), the term **Equijoin** is used.  
<https://eduassistpro.github.io/>

# Example - Equijoin

- List the names and comments of all clients who have viewed a property for rent.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

client.clientNo	fName	lName	ViewIn	PropertyNo	comment
CR76	John	Kay	CR76	PG4	too remote
CR56	Aline	Stewart	CR56	PA14	too small
CR56	Aline	Stewart	CR56	PG4	
CR56	Aline	Stewart	CR56	PG36	
CR62	Mary	Tregear	CR62	PA14	no dining room

# Natural Join

- $R \bowtie S$ 
  - An Equijoin of the two relations R and S over all common attributes  $x$ . One occurrence of each common att

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Example - Natural Join

- List the names and comments of all clients who have viewed a property for rent.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

clientNo	fName	lName		comment
CR76	John	Kay		too remote
CR56	Aline	Stewart	PA14	too small
CR56	Aline	Stewart	PG4	
CR56	Aline	Stewart	PG36	
CR62	Mary	Tregear	PA14	no dining room

# Outer Join

- To display rows in the result that do not have matching values in the join column, use Outer join.
- $R \text{ } \bowtie \text{ } S$       Assignment Project Exam Help
  - (Left) outer join from R that do not have matching values columns of S are also included in result

# Example - Left Outer Join

- Produce a status report on property viewings.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

propertyNo	street	city		viewDate	comment
PA14	16 Holhead	Aberdeen	Add WeChat edu_assist_pro	May-01	too small
PA14	16 Holhead	Aberdeen	CR62	14-May-01	no dining room
PL94	6 Argyll St	London	null	null	null
PG4	6 Lawrence St	Glasgow	CR76	20-Apr-01	too remote
PG4	6 Lawrence St	Glasgow	CR56	26-May-01	
PG36	2 Manor Rd	Glasgow	CR56	28-Apr-01	
PG21	18 Dale Rd	Glasgow	null	null	null
PG16	5 Novar Dr	Glasgow	null	null	null

# Semijoin

- $R \triangleright_F S$ 
  - Defines a relation that contains the tuples of R that participate in the join of R with S.

- Can rewrite  $\text{Se}$  <https://eduassistpro.github.io/>

$$R \triangleright_F S = \Pi_A(R \bowtie_F S)$$

# Example - Semijoin

- List complete details of all staff who work at the branch in Glasgow.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

staffNo	fName	IName	position	s		salary	branchNo
SG37	Ann	Beech	Assistant	F		12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003

# Division

- $R \div S$ 
  - Defines a relation over the attributes C that consists of set of tuples from R that match combination of **every tuple** <https://eduassistpro.github.io/>
- Expressed using:
$$T_1 \leftarrow \prod_c (R)$$
$$T_2 \leftarrow \prod_c ((S \times T_1) - R)$$
$$T \leftarrow T_1 - T_2$$

# Example - Division

- Identify all clients who have viewed all properties with three rooms.

$(\Pi_{clientNo, propertyNo} (Viewing))$

$(\Pi_{propertyNo} (\sigma_{rooms=3} (PropertyForRent)))$

$(\Pi_{clientNo, propertyNo} (Viewing))$

Add WeChat edu\_assist\_pro

Result

clientNo	propertyNo
CR56	PA14
CR76	PG4
CR56	PG4
CR62	PA14
CR56	PG36

$(\Pi_{propertyNo} (\sigma_{rooms=3} (PropertyForRent)))$

propertyNo
PG4
PG36

clientNo
CR56

# Aggregate Operations

- $\mathfrak{I}_{AL}(R)$ 
  - Applies aggregate function list, AL, to R to define a relation over the aggregate list.
  - AL contains <https://eduassistpro.github.io/>, `<function>, <attribute>` pairs.  
**Add WeChat edu\_assist\_pro**
- Main aggregate functions are: COUNT, SUM, AVG, MIN, and MAX.

# Example – Aggregate Operations

- How many properties cost more than £350 per month to rent?

$\rho_R(\text{myCount})$  Assignment Project Exam Help  
nt))

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

myCount
5

(a)

# Grouping Operation

- $\text{GA } \mathfrak{T}_{\text{AL}}(R)$ 
  - Groups tuples of R by grouping attributes, GA, and then applies **Assignment Project Exam Help AL**, to define a new relation <https://eduassistpro.github.io/>
  - A L contains one or more ( $\text{function}$ ,  $\langle \text{attribute} \rangle$ ) pairs.
  - Resulting relation contains the grouping attributes, GA, along with results of each of the aggregate functions.

# Example – Grouping Operation

- Find the number of staff working in each branch and the sum of their salaries.

$\rho_R$  (branchNo, myCount, mySum)

branchNo  $\Sigma$  COUNT st  
<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

branchNo	myCount	
B003	3	54000
B005	2	39000
B007	1	9000

# Relational Calculus (1 of 2)

- Relational calculus query specifies **what** is to be retrieved rather than **how** to retrieve it.
  - No description of how to evaluate a query.
- In first-order logic, <https://eduassistpro.github.io/>, **predicate** is a truth-valued function.
- When we substitute values for variables, function yields an expression, called a **proposition**, which can be either true or false.

# Relational Calculus (2 of 2)

- If predicate contains a variable (e.g. ‘x is a member of staff’), there must be a range for x.
- When we substitute some values of this range for x, proposition may be false.  
<https://eduassistpro.github.io/>
- When applied to databases, relational calculus has forms: **tuple** and **domain**.  
[Add WeChat edu\\_assist\\_pro](#)

# Tuple Relational Calculus (1 of 6)

- Interested in finding tuples for which a predicate is true.  
Based on use of **tuple variables**.
- Tuple variable is a variable that ‘ranges over’ a named relation: i.e., values are tuples of the relation.
- Specify range of a tuple variable as:  
 $\text{Staff}(S)$
- To find set of all tuples S such that  $P(S)$  is true:  
 $\{S \mid P(S)\}$

# Tuple Relational Calculus - Example

- To find details of all staff earning more than £10,000:

$$\{S \mid \text{Staff}(S) \wedge S.\text{salary} > 10000\}$$

Assignment Project Exam Help

- To find a partic lary, write:

<https://eduassistpro.github.io/>  
 $\{S.\text{salary} \mid St$

Add WeChat edu\_assist\_pro

# Tuple Relational Calculus (2 of 6)

- Can use two **quantifiers** to tell how many instances the predicate applies to:
  - Existential quantifier  $\exists$  ('there exists')
  - Universal quantifier  $\forall$
- Tuple variables qualified by  $\forall$  called **bound** variables, otherwise called **free**.

# Tuple Relational Calculus (3 of 6)

- Existential quantifier used in formulae that must be true for at least one instance, such as:

Staff(S)  $\wedge$  ( $\exists B$ )(Branch(B)  $\wedge$  (B.branchNo = S.branchNo)  $\wedge$  B.city = 'London')

<https://eduassistpro.github.io/>

- Means ‘There exists a Branch tuple  $B$  such that  $B$ .branchNo is the branchNo of the current Staff tuple,  $S$ , and is located in London’.

# Tuple Relational Calculus (4 of 6)

- Universal quantifier is used in statements about every instance, such as:

$(\forall B)(B.\text{city} \neq \text{Paris})$

- Means 'For all branches, the city is not in Paris'.  
<https://eduassistpro.github.io/>
- Can also use  $\sim (\exists B)(B.\text{city} = \text{Paris})$  which means  
'There are no branches with a city in Paris'.

# Tuple Relational Calculus (5 of 6)

- Formulae should be unambiguous and make sense.
- A (well-formed) formula is made out of **atoms**:
  - $R(S_i)$ , where  $S_i$  is a tuple variable and  $R$  is a relation
  - $S_i.a_1 \theta S_j.a_2$
  - $S_i.a_1 \theta c$
- Can recursively build up formulae from
  - An atom is a formula
  - If  $F_1$  and  $F_2$  are formulae, so are their conjunction,  $F_1 \wedge F_2$ ; disjunction,  $F_1 \vee F_2$ ; and negation,  $\sim F_1$
  - If  $F$  is a formula with free variable  $X$ , then  $(\exists X)(F)$  and  $(\forall X)(F)$  are also formulae.

## Example - Tuple Relational Calculus (1 of 3)

- List the names of all managers who earn more than £25,000.

Assignment Project Exam Help  
{S.fName, S.lName | Staff(S) ∧  
S.position = 'Manager'}

- List the staff who manage properties in Glasgow.

Add WeChat edu\_assist\_pro  
{S | Staff(S) ∧ (∃P)(PropertyForRent(P) ∧  
(P.StaffNo = S.staffNo) ∧ P.city = 'Glasgow')}

## Example - Tuple Relational Calculus (2 of 3)

- List the names of staff who currently do not manage any properties.

{S fName, S lName | Staff(S)  $\wedge$  ( $\neg (\exists P)$   
(PropertyF <https://eduassistpro.github.io/>)))}

Add WeChat edu\_assist\_pro

Or

{S fName, S lName | Staff(S)  $\wedge$  (( $\forall P$ )  
 $\neg$  PropertyForRent(P)  $\vee$   
 $\neg$  (S staffNo = P staffNo)))}

## Example - Tuple Relational Calculus (3 of 3)

- List the names of clients who have viewed a property for rent in Glasgow.

Assignment Project Exam Help

```
{C.fName,C.lNa  
  (Viewing(V) ∧ Property  
   (C.clientNo = V.clientNo) ∧  
    (V.propertyNo = P.propertyNo) ∧  
    P.city = 'Glasgow'))}
```

# Tuple Relational Calculus (6 of 6)

- Expressions can generate an infinite set. For example:  
 $\{S \mid \sim \text{Staff}(S)\}$
- To avoid this, add restriction that all values in result must be values in the <https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Domain Relational Calculus (1 of 2)

- Uses variables that take values from **domains** instead of tuples of relations.
- If  $F(d_1, d_2, \dots, d_n)$  stands for a formula composed of **Assignment**, **Project**, **Exam**, **Help** composed of **Add**, **WeChat**, **edu\_assist** then:

$\{d_1, d_2, \dots, d_n\} F(d_1, d_2, \dots, d_n)$

is a general domain relational calculus expression.

# Example - Domain Relational Calculus (1 of 4)

- Find the names of all managers who earn more than £25,000.

{fN, IN | ( $\exists$ sN, posn, sex, DOB, sal, bN)}

(Staff(s <https://eduassistpro.github.io/>)  $\wedge$

posn = ‘Manager’  $\wedge$  s }

Add WeChat edu\_assist\_pro

## Example - Domain Relational Calculus (2 of 4)

- List the staff who manage properties for rent in Glasgow.

{sN, fN, IN, posn, sex, DOB, sal, bN |  
    ~~( $\exists$ sN1, cty)(Staff(~~, sal, bN)  $\wedge$   
    PropertyFor ~~Add,~~  $\wedge$  WeChat <https://eduassistpro.github.io/>  
~~s,~~  
    rnt, oN, sN1, bN1)  $\wedge$  ~~Add,~~ WeChat [edu\\_assist\\_pro](https://edu_assist_pro)  
(sN = sN1)  $\wedge$  cty = 'Glasgow')}  
    ~~Assignment Project Exam Help~~

## Example - Domain Relational Calculus (3 of 4)

- List the names of staff who currently do not manage any properties for rent.

{fN, IN | Assignment Project Exam Help  
( $\exists$ sN) (Staff(sN, fN, IN) <https://eduassistpro.github.io/>)  $\wedge$   
 $\neg (\exists$ sN1)(PropertyForRent(pN, sN1, bN1)  $\wedge$  (sN = sN1)))}

# Example - Domain Relational Calculus (4 of 4)

- List the names of clients who have viewed a property for rent in Glasgow.

{fN,IN | ( $\exists cN, cN1, pN, pN1, cty$ )  
(Client(cN, fN, IN) <https://eduassistpro.github.io/>  
Viewing(cN1, pN1, dt, cmt) <sup>^</sup>  
PropertyForRent(pN, st, cty, pc, ty [Add WeChat edu\\_assist\\_pro](#) t, oN, sN, bN)  $\wedge$   
 $(cN = cN1) \wedge (pN = pN1) \wedge cty = 'Glasgow'$ )}

# Domain Relational Calculus (2 of 2)

- When restricted to safe expressions, domain relational calculus is equivalent to tuple relational calculus restricted to safe expressions, which is equivalent to relational algebra
- Means every relation has an equivalent relational calculus expression and vice versa.

# Other Languages (1 of 2)

- Transform-oriented languages are non-procedural languages that use relations to transform input data into required outputs (e.g. SQL). **Assignment Project Exam Help**
- Graphical languages picture of the structure of the wanted and system returns response in that format (e.g. QBE). <https://eduassistpro.github.io/> **Add WeChat edu\_assist\_pro**

# Other Languages (2 of 2)

- 4GLs can create complete customized application using limited set of commands in a user-friendly, often menu-driven environment.
  - Some systems sometimes call development is still at an early stage.
- Assignment Project Exam Help
- <https://eduassistpro.github.io/> I language,  
Add WeChat edu\_assist\_pro

# Copyright

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Database Systems: A Practical Approach to Design, Implementation, and Management

Sixth Edition

Assignment Project Exam**Chapter 6**

<https://eduassistpro.github.io/>  
Data Manipulation

Add WeChat edu\_assist\_pro

# Learning Objectives (1 of 2)

**6.1** Purpose and importance of SQL.

**6.2** How to retrieve data from database using SELECT and:  
**Assignment Project Exam Help**

- Use composition
- Sort query results <https://eduassistpro.github.io/>
- Use aggregate functions [Add WeChat edu\\_assist\\_pro](#)
- Group data using GROUP BY and HAVING.
- Use subqueries.

# Learning Objectives (2 of 2)

- Join tables together.
- Perform set operations (UNION, INTERSECT, EXCEPT)

Assignment Project Exam Help

6.3 How to update, https://eduassistpro.github.io/, UPDATE, and DELETE.

Add WeChat edu\_assist\_pro

# Objectives of SQL (1 of 5)

- Ideally, database language should allow user to:
  - create the database and relation structures;
  - perform insertion, modification, deletion of data from relations;
  - perform simple operations.
- Must perform these tasks with lesser effort and command structure/syntax must be easy to learn.
- It must be portable.

# Objectives of SQL (2 of 5)

- SQL is a transform-oriented language with 2 major components:
  - A DDL for defining database structure.
  - A DML for manipulating data.
- Until SQL:1999, SQL did not have built-in commands. These had to be implemented using a programming or job-control language, or interactively by the decisions of user.

# Objectives of SQL (3 of 5)

- SQL is relatively easy to learn:
  - it is non-procedural - you specify **what** information you require, rather than **how** to get it,
  - it is essenti

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Objectives of SQL (4 of 5)

Consists of standard English words:

- 1) CREATE TABLE Staff(staffNo VARCHAR(5),  
INa  
sala<https://eduassistpro.github.io/>
- 2) INSERT INTO Staff VALUES 'edu\_assist\_pro', 8300);
- 3) SELECT staffNo, IName, salary  
FROM Staff  
WHERE salary > 10000;

# Objectives of SQL (5 of 5)

- Can be used by range of users including DBAs, management, application developers, and other types of end users. **Assignment Project Exam Help**
- An ISO standard making it both the formal and **definitive** standard for relational databases. **Add WeChat edu\_assist\_pro**  
<https://eduassistpro.github.io/>

# History of SQL (1 of 3)

- In 1974, D. Chamberlin (IBM San Jose Laboratory) defined language called ‘Structured English Query Language’ (SEQUEL). [Assignment](#) [Project](#) [Exam](#) [Help](#)
- A revised version was submitted in 1976 but the name was changed to SQL for legal reasons. <https://eduassistpro.github.io/> [Add WeChat edu\\_assist\\_pro](#)

# History of SQL (2 of 3)

- Still pronounced ‘see-quel’, though official pronunciation is ‘S-Q-L’.
- IBM subsequently produced a prototype DBMS called **System R**, bas <https://eduassistpro.github.io/>
- Roots of SQL, however, are in **Queries as Relational Expressions** (Specifying **Add WeChat edu\_assist\_pro** in predicates) in the System R project.

# History of SQL (3 of 3)

- In late 70s, ORACLE appeared and was probably first commercial RDBMS based on SQL.
- In 1987, ANSI and ISO published an initial standard for SQL.  
**Assignment Project Exam Help**
- In 1989, ISO published the 'Integrity Enhancement Feature'.  
**<https://eduassistpro.github.io/>**
- In 1992, first major revision to ISO standard referred to as SQL2 or SQL/92.  
**Add WeChat edu\_assist\_pro**
- In 1999, SQL:1999 was released with support for object-oriented data management.
- In late 2003, SQL:2003 was released.
- In summer 2008, SQL:2008 was released.
- In late 2011, SQL:2011 was released.

# Importance of SQL (1 of 2)

- SQL has become part of application architectures such as IBM's Systems Application Architecture.
- It is strategic choice of many large and influential organizations (<https://eduassistpro.github.io/>)
- SQL is Federal Information Pr standard (FIPS) to which conformance is requi Add WeChat `edu_assist_pro` ales of databases to American Government.

# Importance of SQL (2 of 2)

- SQL is used in other standards and even influences development of other standards as a definitional tool.

Examples include:

- ISO's Information **Standard** **Assignment Project Exam Help** System (IRDS) <https://eduassistpro.github.io/>
- Remote Data Access (RDA) **Add WeChat** **edu\_assist\_pro**

# Writing SQL Commands (1 of 3)

- SQL statement consists of **reserved words** and **user-defined words**.
  - Reserved words are ~~Assignment Project Exam Help~~ fixed part of SQL and must be spelt exactly. <https://eduassistpro.github.io/>
  - User-defined words are used to represent names of various objects such as relations, columns, views.

# Writing SQL Commands (2 of 3)

- Most components of an SQL statement are **case insensitive**, except for literal character data.
- More readable with indentation and lineation:
  - Each clause starts on a new line.
  - Start of a clause should line up with start of other clauses. Add WeChat edu\_assist\_pro
  - If clause has several parts, should each appear on a separate line and be indented under start of clause.

# Writing SQL Commands (3 of 3)

- Use extended form of BNF notation:
  - Upper-case letters represent reserved words.
  - Lower-case letters represent user-defined words.
  - | indicates a <https://eduassistpro.github.io/> ives.
  - Curly brace t.
  - Square brackets indicate [Add WeChat edu\\_assist\\_pro element.](#)
  - ... indicates **optional repetition** (0 or more).

# Literals

- Literals are constants used in SQL statements.
- All non-numeric literals must be enclosed in single quotes (e.g. 'London').
- All numeric literals must be enclosed in double quotes (e.g. 650.00).

Add WeChat edu\_assist\_pro

# **SELECT Statement (1 of 3)**

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# SELECT Statement (2 of 3)

FROM	Specifies table(s) to be used.
WHERE	Filters rows.
GROUP BY	Forms groups of rows with same column val <a href="https://eduassistpro.github.io/">https://eduassistpro.github.io/</a>
HAVING	Filter me condition.
SELECT	Add WeChat <code>edu_assist_pro</code> to appear in Specifies which c output.
ORDER BY	Specifies the order of the output.

# SELECT Statement (3 of 3)

- Order of the clauses cannot be changed.
- Only SELECT and FROM are mandatory.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

## Example 6.1 All Columns, All Rows (1 of 2)

List full details of all staff.

```
SELECT staffNo, fName, lName, address,  
position, sex,  
FROM Staff;
```

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- Can use \* as an abbreviation for ‘all columns’:

```
SELECT *  
FROM Staff;
```

## Example 6.1 All Columns, All Rows (2 of 2)

staffNo	fName	IName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000.00	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000.00	B003
SG14	David	Fo	Manager	M	15-Aug-58	18000.00	B003
SA9	Mary	Howe	Assistant	F		9000.00	B007
SG5	Susan	Brand	Manager	M		24000.00	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000.00	B005

## Example 6.2 Specific Columns, All Rows (1 of 2)

Produce a list of salaries for all staff, showing only staff number, first and last names, and salary.

Assignment Project Exam Help

SELECT st <https://eduassistpro.github.io/>

FROM Staff;

Add WeChat edu\_assist\_pro

## Example 6.2 Specific Columns, All Rows (2 of 2)

staffNo	fName	lName	salary
SL21	John	White	30000.00
SG37	Ann	Beech	12000.00
SG14			18000.00
SA9			9000.00
SG5	Susan	Br	4000.00
SL41	Julie	L	9000.00

## Example 6.3 Use of DISTINCT (1 of 2)

List the property numbers of all properties that have been viewed.

Assignment Project Exam Help

```
SELECT p  
FROM Vie
```

<https://eduassistpro.github.io/>

tyNo
PG4
PG4
PA14
PG36

## Example 6.3 Use of DISTINCT (2 of 2)

- Use DISTINCT to eliminate duplicates:

```
Assignment Project Exam Help  
SELECT DISTINCT propertyNo  
FROM View https://eduassistpro.github.io/
```

propertyNo
PA14
PG4
PG36

## Example 6.4 Calculated Fields (1 of 2)

Produce list of monthly salaries for all staff, showing staff number, first/last name, and salary.

Assignment Project Exam Help  
SELECT staffNo, fName, lName, salary/12  
FROM Staff

<https://eduassistpro.github.io/>

staffNo	fName	lName	salary/12
SL21	John	White	2500.00
SG37	Ann	Beech	1000.00
SG14	David	Ford	1500.00
SA9	Mary	Howe	750.00
SG5	Susan	Brand	2000.00
SL41	Julie	Lee	750.00

## Example 6.4 Calculated Fields (2 of 2)

- To name column, use AS clause:

```
Assignment Project Exam Help  
SELECT staffNo, fName, lName, salary/12  
      https://eduassistpro.github.io/  
FROM Staff; Add WeChat edu_assist_pro
```

## Example 6.5 Comparison Search Condition

List all staff with a salary greater than 10,000.

```
SELECT staffNo, fName, lName, position, salary  
FROM Staff
```

```
WHERE sal > 10000
```

staffNo	fName	lName	position	salary
SL21	John	White	Manager	30000.00
SG37	Ann	Beech	Assistant	12000.00
SG14	David	Ford	Supervisor	18000.00
SG5	Susan	Brand	Manager	24000.00

# Example 6.6 Compound Comparison Search Condition

List addresses of all branch offices in London or Glasgow.

```
SELECT *  
FROM Branch  
WHERE city = 'London' OR city = 'Glasgow';
```

Add WeChat edu\_assist\_pro

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B003	163 Main St	Glasgow	G11 9QX
B002	56 Clover Dr	London	NW10 6EU

## Example 6.7 Range Search Condition (1 of 3)

List all staff with a salary between 20,000 and 30,000.

SELECT Assignment, Project, Exam, Help, salary

FROM Staff

WHERE salary BETWEEN 20000 AND 30000;

<https://eduassistpro.github.io/>  
Add WeChat edu\_assist\_pro

- BETWEEN test includes the endpoints of range.

## Example 6.7 Range Search Condition (2 of 3)

staffNo	fName	IName	position	salary
SL21	John	White	Manager	30000.00
SG5	Susan	Brand	Manager	24000.00

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

## Example 6.7 Range Search Condition (3 of 3)

- Also a negated version NOT BETWEEN.
- BETWEEN does not add much to SQL's expressive power. Could also write:

```
Assignment Project Exam Help  
SELECT st https://eduassistpro.github.io/  
        sition, salary  
FROM StaffAdd WeChat edu_assist_pro  
WHERE salary>=20000 AND salary <= 30000;
```

- Useful, though, for a range of values.

## Example 6.8 Set Membership (1 of 2)

List all managers and supervisors.

```
SELECT staffNo, fName, lName, position  
FROM Staff  
WHERE position = 'Manager' OR position = 'Supervisor';
```

<https://eduassistpro.github.io/>  
Add WeChat edu\_assist\_pro

staffNo	fName	lName	position
SL21	John	White	Manager
SG14	David	Ford	Supervisor
SG5	Susan	Brand	Manager

## Example 6.8 Set Membership (2 of 2)

- There is a negated version (NOT IN).
- IN does not add much to SQL's expressive power. Could have expressed this as.

<https://eduassistpro.github.io/>

```
SELECT st position
FROM StaffAdd WeChat edu_assist_pro
WHERE position='Manager' OR
      position='Supervisor';
```

- IN is more efficient when set contains many values.

## Example 6.9 Pattern Matching (1 of 2)

Find all owners with the string ‘Glasgow’ in their address.

```
SELECT ownerNo, fName, lName, address, telNo  
FROM PrivateO  
WHERE address
```

<https://eduassistpro.github.io/>  
Add WeChat edu\_assist\_pro

ownerNo	fName	lName	address	telNo
CO87	Carol	Farrel	6 Achray St, Glasgow G32 9DX	0141-357-7419
CO40	Tina	Murphy	63 Well St, Glasgow G42	0141-943-1728
CO93	Tony	Shaw	12 Park PI, Glasgow G4 0QR	0141-225-7025

## Example 6.9 Pattern Matching (2 of 2)

- SQL has two special pattern matching symbols:
  - %: sequence of zero or more characters;
  - \_: (underscore): any single character.
- LIKE '%Glasgo' <https://eduassistpro.github.io/> of characters of any length containing 'Glasgo'  
AddWeChat  
[https://edu\\_assist\\_pro](https://edu_assist_pro)

## Example 6.10 NULL Search Condition (1 of 2)

List details of all viewings on property PG4 where a comment has not been supplied.

- There are 2 viewings for property PG4, one with and one without a comm

<https://eduassistpro.github.io/>

- Have to test for null explicitly

al keyword IS  
NULL:

```
SELECT clientNo, viewDate
FROM Viewing
WHERE propertyNo = 'PG4' AND
      comment IS NULL;
```

## Example 6.10 NULL Search Condition (2 of 2)

clientNo	viewDate
CR 56	26-May-13

Assignment Project Exam Help

- Negated version <https://eduassistpro.github.io/> st for non-null values.

Add WeChat edu\_assist\_pro

## Example 6.11 Single Column Ordering (1 of 2)

List salaries for all staff, arranged in descending order of salary.

Assignment Project Exam Help

```
SELECT st          lary
FROM Staff        https://eduassistpro.github.io/
ORDER BY AddWeChat; edu_assist_pro
```

## Example 6.11 Single Column Ordering (2 of 2)

staffNo	fName	IName	salary
SL21	John	White	30000.00
SG5			4000.00
SG14			8000.00
SG37	Ann		00.00
SA9	Mary	Howe	9000.00
SL41	Julie	Lee	9000.00

## Example 6.12 Multiple Column Ordering (1 of 4)

Produce abbreviated list of properties in order of property type.

```
Assignment Project Exam Help  
SELECT pr          rent  
FROM Prophttps://eduassistpro.github.io/  
ORDER BY type;  
Add WeChat edu_assist_pro
```

## Example 6.12 Multiple Column Ordering (2 of 4)

propertyNo	type	rooms	rent
PL94	Flat	4	400
PG4			350
PG36			375
PG16	Flat		50
PA14	House		50
PG21	House	5	600

## Example 6.12 Multiple Column Ordering (3 of 4)

- Four flats in this list - as no minor sort key specified, system arranges these rows in any order it chooses.
- To arrange in order of rent, specify minor order:

<https://eduassistpro.github.io/>

```
SELECT propertyNo, type          nt  
      ,  
      Add WeChat edu_assist_pro  
FROM PropertyForRent  
ORDER BY type, rent DESC;
```

## Example 6.12 Multiple Column Ordering (4 of 4)

propertyNo	type	rooms	rent
PL16	Flat	4	450
PG94			400
PG36			375
PG4	Flat		50
PA14	House		50
PG21	House	5	600

# SELECT Statement – Aggregates (1 of 4)

- ISO standard defines five aggregate functions:

COUNT returns number of values in specified column.  
[Assignment](#) [Project](#) [Exam](#) [Help](#)

SUM returns sum column.

<https://eduassistpro.github.io/>

AVG returns average column.

[Add WeChat](#) [edu\\_assist\\_pro](#)

MIN returns smallest value in specified column.

MAX returns largest value in specified column.

# SELECT Statement – Aggregates (2 of 4)

- Each operates on a single column of a table and returns a single value.
- COUNT, MIN, and MAX apply to numeric and non-numeric fields, <https://eduassistpro.github.io/> be used on numeric fields o
- Apart from COUNT(\*), each function ignores nulls first and operates only on remaining non-null values.

# SELECT Statement – Aggregates (3 of 4)

- COUNT(\*) counts all rows of a table, regardless of whether nulls or duplicate values occur.
- Can use DISTINCT before column name to eliminate duplicates. <https://eduassistpro.github.io/>
- DISTINCT has no effect with SUM/AVG. [Add WeChat edu\\_assist\\_pro](#)

# SELECT Statement – Aggregates (4 of 4)

- Aggregate functions can be used only in SELECT list and in HAVING clause.
- If SELECT list includes an aggregate function and there is no GROUP BY clause, you cannot reference a column outside of the group. For example, the following is illegal:  
<https://eduassistpro.github.io/>  
Assignment Project Exam Help  
Add WeChat edu\_assist\_pro

```
SELECT staffNo, COUNT(salary)  
FROM Staff;
```

## Example 6.13 Use of COUNT(\*)

- How many properties cost more than £350 per month to rent?

Assignment Project Exam Help

```
SELECT C
FROM Prop
WHERE rent > 350;
```

myCount
5

## Example 6.14 Use of COUNT(DISTINCT)

- How many different properties viewed in May '13?

```
SELECT COUNT(DISTINCT propertyNo) AS myCount  
FROM View  
WHERE vie https://eduassistpro.github.io/ay-13  
      AND '21-May-13'  
      AND WeChat edu_assist_pro
```

myCount
2

## Example 6.15 Use of COUNT and SUM

- Find number of Managers and sum of their salaries.

```
SELECT COUNT(staffNo) AS myCount  
      , SUM(salary)          AS mySum  
   FROM Staff  
 WHERE position = 'Manager'  
https://eduassistpro.github.io/
```

myCount	mySum
2	54000.00

## Example 6.16 Use of MIN, MAX, AVG

- Find minimum, maximum, and average staff salary.

```
SELECT MIN(salary) AS myMin,  
       Assignment Project Exam Help  
       MAX(  
       AVG(  
FROM Staff;Add WeChat edu_assist_pro
```

myMin	myMax	myAvg
9000.00	30000.00	17000.00

# SELECT Statement – Grouping (1 of 2)

- Use GROUP BY clause to get sub-totals.
- SELECT and GROUP BY closely integrated: each item in SELECT list must be **single-valued per group**, and  
SELECT clause
  - column names
  - aggregate functions
  - constants
  - expression involving combinations of the above.

## SELECT Statement – Grouping (2 of 2)

- All column names in SELECT list must appear in GROUP BY clause unless name is used only in an aggregate function. [Assignment](#) [Project](#) [Exam](#) [Help](#)
- If WHERE is used first, then group satisfying predicate. <https://eduassistpro.github.io/> HERE is applied ining rows
- ISO considers two nulls to be equal for purposes of GROUP BY.

## Example 6.17 Use of GROUP BY (1 of 2)

- Find number of staff in each branch and their total salaries.

```
SELECT branchNo, COUNT(staffID) AS staffCount, SUM(salary) AS totalSalary  
FROM Staff  
GROUP BY branchNo  
ORDER BY branchNo;
```

## Example 6.17 Use of GROUP BY (2 of 2)

branchNo	myCount	mySum
B003	Assignment Project Exam Help 3	94000.00
B005		0.00
B007	https://eduassistpro.github.io/	0.00

Add WeChat edu\_assist\_pro

# Restricted Groupings – HAVING Clause

- HAVING clause is designed for use with GROUP BY to restrict groups that appear in final result table.
- Similar to WHERE, but WHERE filters individual rows whereas HAVING <https://eduassistpro.github.io/>
- Column names in HAVING clause must appear in the GROUP BY list or be contained in an aggregate function.

## Example 6.18 Use of HAVING (1 of 2)

- For each branch with more than 1 member of staff, find number of staff in each branch and sum of their salaries.

Assignment Project Exam Help

```
SELECT br  
    COU https://eduassistpro.github.io/  
        SUM(salary) AS my_edu_assist_pro  
FROM Staff  
GROUP BY branchNo  
HAVING COUNT(staffNo) > 1  
ORDER BY branchNo;
```

## Example 6.18 Use of HAVING (2 of 2)

Assignment Project Exam Help		
branchNo	myCount	mySum
B003		00.00
B005	<a href="https://eduassistpro.github.io/">https://eduassistpro.github.io/</a>	00.00

Add WeChat edu\_assist\_pro

# Subqueries

- Some SQL statements can have a SELECT embedded within them.
- A subselect can be used in WHERE and HAVING clauses of an o s called a <https://eduassistpro.github.io/>  
**subquery** or n
- Subselects may also appear i UPDATE, and DELETE statements.

## Example 6.19 Subquery with Equality (1 of 3)

- List staff who work in branch at ‘163 Main St’.

```
SELECT StaffNo, lName, fName, position  
FROM Staff  
WHERE br  
    (SELECT branchNo  
     FROM Branch  
     WHERE street = '163 Main St');
```

## Example 6.19 Subquery with Equality (2 of 3)

- Inner SELECT finds branch number for branch at ‘163 Main St’ (‘B003’).
- Outer SELECT then retrieves details of all staff who work at this branch.

<https://eduassistpro.github.io/>

- Outer SELECT then becomes

[Add WeChat edu\\_assist\\_pro](#)

```
SELECT staffNo, fName, lName, position  
FROM Staff  
WHERE branchNo = 'B003';
```

## Example 6.19 Subquery with Equality (3 of 3)

staffNo	fName	lName	position
SG37			Assistant
SG14			Supervisor
SG5	Susan	Br	Manager

**Add WeChat edu\_assist\_pro**

## Example 6.20 Subquery with Aggregate (1 of 3)

List all staff whose salary is greater than the average salary, and show by how much.

Assignment Project Exam Help

```
SELECT staff          ition,  
       salary - (SEL https://eduassistpro.github.io/ Staff) As SalDiff  
FROM Staff    Add WeChat edu_assist_pro  
WHERE salary >  
      (SELECT AVG(salary)  
       FROM Staff);
```

## Example 6.20 Subquery with Aggregate (2 of 3)

- Cannot write ‘WHERE salary > AVG(salary)’
- Instead, use subquery to find average salary (17000),  
and then use outer SELECT to find those staff with salary  
greater than thi

<https://eduassistpro.github.io/>

```
SELECT staffNo, fName, lName, address, city, state, zip, phone,  
      salary - 17000 As salDiff  
  FROM Staff  
 WHERE salary > 17000;
```

## Example 6.20 Subquery with Aggregate (3 of 3)

staffNo	fName	IName	position	salDiff
SL21	John	White	Manager	13000.00
SG14	David		visor	1000.00
SG5	Susan			7000.00

Add WeChat edu\_assist\_pro

# Subquery Rules (1 of 2)

- ORDER BY clause may not be used in a subquery (although it may be used in outermost SELECT).
- Subquery SELECT list must consist of a single column name or expression that uses the <https://eduassistpro.github.io/> **EXISTS**.
- By default, column names refer to the same in FROM clause of subquery. Can refer to a table in FROM using an **alias**.

## Subquery Rules (2 of 2)

- When subquery is an operand in a comparison, subquery must appear on right-hand side.
- A subquery may not be used as an operand in an expression.

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Example 6.21 Nested Subquery: Use of IN (1 of 2)

- List properties handled by staff at ‘163 Main St’.

```
SELECT propertyNo, street, city, postcode, type, rooms, rent  
FROM PropertyForRent
```

WHERE staffN <https://eduassistpro.github.io/>

(SELECT staffNo

FROM Staff

WHERE branchNo =

(SELECT branchNo

FROM Branch

WHERE street = ‘163 Main St’));

# Example 6.21 Nested Subquery: Use of IN (2 of 2)

propertyNo	street	city	postcode	type	rooms	rent
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375
PG21	18 Dale	Assignment Project Exam Help <a href="https://eduassistpro.github.io/">https://eduassistpro.github.io/</a>				

Add WeChat edu\_assist\_pro

# ANY and ALL

- ANY and ALL may be used with subqueries that produce a single column of numbers.
- With ALL, condition will only be true if it is satisfied by **all** values produced by subquery. <https://eduassistpro.github.io/>
- With ANY, condition will be true if it is satisfied by **any** values produced by subquery. [Add WeChat edu\\_assist\\_pro](#)
- If subquery is empty, ALL returns true, ANY returns false.
- SOME may be used in place of ANY.

## Example 6.22 Use of ANY/SOME (1 of 2)

- Find staff whose salary is larger than salary of at least one member of staff at branch B003.

Assignment Project Exam Help

```
SELECT staff          ition, salary  
FROM Staff    https://eduassistpro.github.io/  
WHERE salary > SOME  
      (SELECT salary  
       FROM Staff  
       WHERE branchNo = 'B003');
```

## Example 6.22 Use of ANY/SOME (2 of 2)

- Inner query produces set {12000, 18000, 24000} and outer query selects those staff whose salaries are greater than any of the values in this set.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

staffNo	fNa			salary
SL21	John	Add White		30000.00
SG14	David	Ford	Supervisor	18000.00
SG5	Susan	Brand	Manager	24000.00

## Example 6.23 Use of ALL (1 of 2)

- Find staff whose salary is larger than salary of every member of staff at branch B003.

Assignment Project Exam Help

```
SELECT st          sition, salary  
FROM Staff        https://eduassistpro.github.io/  
WHERE salary > ALL  
      (SELECT salary  
       FROM Staff  
      WHERE branchNo = 'B003');
```

## Example 6.23 Use of ALL (2 of 2)

staffNo	fName	lName	position	salary
SL21	John			30000.00

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# Multi-Table Queries (1 of 2)

- Can use subqueries provided result columns come from same table.
- If result columns come from more than one table must use a join.  
<https://eduassistpro.github.io/>
- To perform join, include more table in FROM clause.  
[Add WeChat edu\\_assist\\_pro](#)
- Use comma as separator and typically include WHERE clause to specify join column(s).

# Multi-Table Queries (2 of 2)

- Also possible to use an alias for a table named in FROM clause.
- Alias is separated from table name with a space.
- Alias can be used when there is ambiguity.

Add WeChat edu\_assist\_pro

## Example 6.24 Simple Join (1 of 2)

- List names of all clients who have viewed a property along with any comment supplied.

Assignment Project Exam Help

```
SELECT c.cli https://eduassistpro.github.io/
      property
FROM Client c, Viewing v
          Add WeChat edu_assist_pro
WHERE c.clientNo = v.clientNo;
```

## Example 6.24 Simple Join (2 of 2)

- Only those rows from both tables that have identical values in the clientNo columns ( $c.clientNo = v.clientNo$ ) are included in result.
- Equivalent to eq

Assignment Project Exam Help

<https://eduassistpro.github.io/>

clientNo	fNa			comment
CR56	Aline	Stewart	P	Add WeChat edu_assist_pro
CR56	Aline	Stewart	PA14	too small
CR56	Aline	Stewart	PG4	
CR62	Mary	Tregear	PA14	no dining room
CR76	John	Kay	PG4	too remote

# Alternative JOIN Constructs

- SQL provides alternative ways to specify joins:

```
FROM Client c JOIN Viewing v ON c.clientNo = v.clientNo  
Assignment Project Exam Help  
FROM Client J ntNo  
FROM Client N https://eduassistpro.github.io/
```

- In each case, FROM replaces ~~OM~~ and WHERE. However, first produces table with two identical clientNo columns.

## Example 6.25 Sorting a Join (1 of 2)

- For each branch, list numbers and names of staff who manage properties, and properties they manage.

Assignment Project Exam Help

```
SELECT s.branchNo, s.IName,  
       propertyNo  
FROM Staff s, PropertyForR  
WHERE s.staffNo = p.staffNo  
ORDER BY s.branchNo, s.staffNo, propertyNo;
```

## Example 6.25 Sorting a Join (2 of 2)

branchNo	staffNo	fName	lName	propertyNo
B003	SG14	David	Ford	PG16
B003	SG			PG21
B003	SG			PG36
B005	SL41	Julie		PL94
B007	SA9	Mary		PA14

## Example 6.26 Three Table Join (1 of 2)

- For each branch, list staff who manage properties, including city in which branch is located and properties they manage.

Assignment Project Exam Help

```
SELECT b.b  
      , fName, lName,  
      propertyN  
      , https://eduassistpro.github.io/  
      , AddWeChat  
      , edu_assist_pro  
      , t_p  
      , FROM Branch b  
      , Staff s  
      , Property p  
      , WHERE b.branchNo = s.branchNo AND  
            s.staffNo = p.staffNo  
      , ORDER BY b.branchNo, s.staffNo, propertyNo;
```

## Example 6.26 Three Table Join (2 of 2)

branchNo	city	staffNo	fName	IName	propertyNo
B003	Glasgow	SG14	David	Ford	PG16
B003	Glasgow	SG37	Ann	Beech	PG21
B003	Glasgow	SG37	Ann	Beech	PG36
B005	London				PL94
B007	Aberdeen	SA9	M	e	PA14

Add WeChat edu\_assist\_pro

- Alternative formulation for FROM and WHERE:

FROM (Branch b JOIN Staff s USING branchNo) AS  
bs JOIN PropertyForRent p USING staffNo

## Example 6.27 Multiple Grouping Columns (1 of 2)

- Find number of properties handled by each staff member.

```
SELECT s.branchNo, s.staffNo, COUNT(*) AS  
      myCount  
  FROM Staff s https://eduassistpro.github.io/  
 WHERE s.staffNo = p.staffN Add WeChat edu_assist_pro  
 GROUP BY s.branchNo, s.staffNo  
 ORDER BY s.branchNo, s.staffNo;
```

## Example 6.27 Multiple Grouping Columns (2 of 2)

branchNo	staffNo	myCount
B003	Assignment Project Exam Help SQ10	1
B003		
B005	https://eduassistpro.github.io/	
B007	SA9 Add WeChat edu_assist_pro	

# Computing a Join (1 of 2)

Procedure for generating results of a join are:

1. Form Cartesian product of the tables named in FROM clause.
2. If there is a WHERE condition to each row of the product, add it to each row of the product to those rows that satisfy the condition.
3. For each remaining row, determine value of each item in SELECT list to produce a single row in result table.

# Computing a Join (2 of 2)

4. If DISTINCT has been specified, eliminate any duplicate rows from the result table.
  5. If there is an ORDER BY clause, sort result table as required.
- Assignment Project Exam Help
- <https://eduassistpro.github.io/>
- SQL provides special format to Cartesian product:

```
SELECT [DISTINCT | ALL] { * | columnList }
FROM Table1 CROSS JOIN Table2
```

# Outer Joins (1 of 3)

- If one row of a joined table is unmatched, row is omitted from result table.
- Outer join operations retain rows that do not satisfy the join condition.
- Consider following

<https://eduassistpro.github.io/>

Branch 1

branchNo	bCity
B003	Glasgow
B004	Bristol
B002	London

Add WeChat [edu\\_assist\\_pro](https://eduassistpro.github.io/)

propertyNo	pCity
PA14	Aberdeen
PL94	London
PG4	Glasgow

# Outer Joins (2 of 3)

- The (inner) join of these two tables:

```
SELECT b.* , p.*  
      Assignment Project Exam Help  
      FROM Branc          p  
      WHERE b.bC https://eduassistpro.github.io/
```

Add WeChat edu\_assist\_pro

branchNo	bCity	PropertyNo	pCity
B003	Glasgow	PG4	Glasgow
B002	London	PL94	London

## Outer Joins (3 of 3)

- Result table has two rows where cities are same.
- There are no rows corresponding to branches in Bristol and Aberdeen.
- To include unm  
<https://eduassistpro.github.io/>an Outer join.

Add WeChat edu\_assist\_pro

## Example 6.28 Left Outer Join (1 of 2)

- List branches and properties that are in same city along with any unmatched branches.

Assignment Project Exam Help

```
SELECT b.*,
       https://eduassistpro.github.io/
  FROM Branches b
    LEFT OUTER JOIN Properties p ON b.BranchID = p.BranchID
      WHERE p.City = 'San Francisco'
```

## Example 6.28 Left Outer Join (2 of 2)

- Includes those rows of first (left) table unmatched with rows from second (right) table.
- Columns from second table are filled with NULLs.

<https://eduassistpro.github.io/>

branchNo	bCity	p	City
B003	Glasgow	P	glasgow
B004	Bristol	NULL	NULL
B002	London	PL94	London

## Example 6.29 Right Outer Join (1 of 2)

- List branches and properties in same city and any unmatched properties.

Assignment Project Exam Help

```
SELECT b.*,
       https://eduassistpro.github.io/
  FROM Branch
 PropertyForRent1 p ON Add WeChat edu_assist_pro
          pCity,
```

## Example 6.29 Right Outer Join (2 of 2)

- Right Outer join includes those rows of second (right) table that are unmatched with rows from first (left) table.
- Columns from first table are filled with NULLs.

<https://eduassistpro.github.io/>

branchNo	bCity	p	City
NULL	NULL	P	berdeen
B003	Glasgow	PG4	Glasgow
B002	London	PL94	London

## Example 6.30 Full Outer Join (1 of 2)

- List branches and properties in same city and any unmatched branches or properties.

Assignment Project Exam Help

```
SELECT b.*,
       https://eduassistpro.github.io/
  FROM Branches b
    LEFT JOIN Properties p ON b.BranchID = p.PropertyID
    WHERE b.BranchID = p.PropertyID;
```

## Example 6.30 Full Outer Join (2 of 2)

- Includes rows that are unmatched in both tables.
- Unmatched columns are filled with NULLs.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

branchNo			pCity
NULL	NULL	P	berdeen
B003	Glasgow	P	glasgow
B004	Bristol	NULL	NULL
B002	London	PL94	London

# EXISTS and NOT EXISTS (1 of 2)

- EXISTS and NOT EXISTS are for use only with subqueries.
- Produce a simple true/false result.
- True if and only <https://eduassistpro.github.io/> result table returned by subquery.  
[Assignment](#) [Project](#) [Exam](#) [Help](#)  
[Add WeChat edu\\_assist\\_pro](#)
- False if subquery returns an empty table.
- NOT EXISTS is the opposite of EXISTS.

## EXISTS and NOT EXISTS (2 of 2)

- As (NOT) EXISTS check only for existence or non-existence of rows in subquery result table, subquery can contain any number of columns.  
<https://eduassistpro.github.io/>
- Common form:  
$$(SELECT *...) \text{ EXISTS}$$

## Example 6.31 Query using EXISTS (1 of 4)

- Find all staff who work in a London branch.

```
SELECT StaffID, fName, lName, Position  
FROM Staff s  
WHERE EXI https://eduassistpro.github.io/  
      (SELECT *  
       FROM Branch b  
       WHERE s.branchNo = b.branchNo AND  
             city = 'London');
```

## Example 6.31 Query using EXISTS (2 of 4)

staffNo	Assignment	Project	Exam	Help	position
SL21					Manager
SL41		<a href="https://eduassistpro.github.io/">https://eduassistpro.github.io/</a>			Assistant

Add WeChat edu\_assist\_pro

## Example 6.31 Query using EXISTS (3 of 4)

- Note, search condition `s.branchNo = b.branchNo` is necessary to consider correct branch record for each member of staff
- If omitted, would be:  
`SELECT * FROM Staff WHERE true;`  
Assignment Project Exam Help  
<https://eduassistpro.github.io/>  
Add WeChat edu\_assist\_pro  
y='London'
- would always be true and query would be:  
`SELECT staffNo, fName, lName, position FROM Staff WHERE true;`

## Example 6.31 Query using EXISTS (4 of 4)

- Could also write this query using join construct:

```
Assignment Project Exam Help  
SELECT staffNo, fName, lName, position  
FROM Staff  
WHERE s.branchNo = b.b  
      ND  
      Add WeChat edu_assist_pro  
      city = 'London';
```

# Union, Intersect, and Difference (Except) (1 of 3)

- Can use normal set operations of Union, Intersection, and Difference to combine results of two or more queries into a single result table.
- Union of two tables containing all rows in either A or B
- Intersection is table containing common to both A and B.
- Difference is table containing all rows in A but not in B.
- Two tables must be **union compatible**.

# Union, Intersect, and Difference (Except) (2 of 3)

- Format of set operator clause in each case is:

op [ALL] [CORRESPONDING [BY {column1 [, ...]}]]

Assignment Project Exam Help

- If CORRESPONDING BY specified, set operation performed on t <https://eduassistpro.github.io/>
- If CORRESPONDING specific Y clause, operation performed on comm s.
- If ALL specified, result can include duplicate rows

# Union, Intersect, and Difference (Except) (3 of 3)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

## Example 6.32 Use of UNION (1 of 3)

- List all cities where there is either a branch office or a property.

Assignment Project Exam Help

```
(SELECT ci https://eduassistpro.github.io/  
FROM Bran  
WHERE city IS NOT NUL Add WeChat edu_assist_pro  
(SELECT city  
FROM PropertyForRent  
WHERE city IS NOT NULL);
```

## Example 6.32 Use of UNION (2 of 3)

- Or

```
(SELECT *  
FROM Branch  
WHERE city  
UNION CORRESPONDING  
(SELECT *  
FROM PropertyForRent  
WHERE city IS NOT NULL);
```

## Example 6.32 Use of UNION (3 of 3)

- Produces result tables from both queries and merges both tables together.

Assignment Project Exam Help

<a href="https://eduassistpro.github.io/">https://eduassistpro.github.io/</a>
London
Add WeChat edu_assist_pro
Glasgow
Aberdeen
Bristol

## Example 6.33 Use of INTERSECT (1 of 3)

- List all cities where there is both a branch office and a property.

Assignment Project Exam Help

```
(SELECT ci  
INTERSEC  
(SELECT city FROM Prop);  
https://eduassistpro.github.io/  
Add WeChat edu\_assist\_pro
```

## Example 6.33 Use of INTERSECT (2 of 3)

- Or

```
(SELECT * FROM Branch) Assignment Project Exam Help  
INTERSEC Y city  
(SELECT * https://eduassistpro.github.io/ );
```

Add WeChat edu\_assist\_pro

city
Aberdeen
Glasgow
London

## Example 6.33 Use of INTERSECT (3 of 3)

- Could rewrite this query without INTERSECT operator:

```
SELECT b.city  
      Assignment Project Exam Help  
     FROM Branch  
 WHERE b.chttps://eduassistpro.github.io/
```

- Or: Add WeChat edu\_assist\_pro

```
SELECT DISTINCT city FROM Branch b  
 WHERE EXISTS  
   (SELECT * FROM PropertyForRent p  
    WHERE p.city = b.city);
```

## Example 6.34 Use of EXCEPT (1 of 2)

- List of all cities where there is a branch office but no properties.

(SELECT city FROM Branch)

EXCEPT https://eduassistpro.github.io/  
(SELECT city FROM Prop t);  
Add WeChat edu\_assist\_pro

- Or

(SELECT \* FROM Branch)

EXCEPT CORRESPONDING BY city

(SELECT \* FROM PropertyForRent);

city
Bristol

## Example 6.34 Use of EXCEPT (2 of 2)

- Could rewrite this query without EXCEPT:

```
SELECT DISTINCT city FROM Branch  
Assignment Project Exam Help  
WHERE cit  
(SELEC https://eduassistpro.github.io/  
rRent),
```

- Or Add WeChat edu\_assist\_pro

```
SELECT DISTINCT city FROM Branch b  
WHERE NOT EXISTS  
(SELECT * FROM PropertyForRent p  
WHERE p.city = b.city);
```

# INSERT (1 of 2)

```
INSERT INTO TableName [ (columnList) ]  
VALUES (dataValueList)
```

Assignment Project Exam Help

- **columnList** is a list of all columns in the table. <https://eduassistpro.github.io/> assumes a list of BEE order.
- Any columns omitted must have been declared as NULL when table was created, unless DEFAULT was specified when creating column.

## INSERT (2 of 2)

- **dataValueList** must match **columnList** as follows:
  - number of items in each list must be same;
  - must be direct correspondence in position of items in two lists;
  - data type of <https://eduassistpro.github.io/> List must be compatible with data type ~~Add WeChat~~ ~~edu\_assist\_pro~~ column.

## Example 6.35 INSERT ... VALUES

- Insert a new row into Staff table supplying data for all columns.

Assignment Project Exam Help  
INSERT INTO  
VALUES ('SGhttps://eduassistpro.github.io/Assistant', 'M',  
Date '1957-05-25', 830  
Add WeChat edu\_assist\_pro

## Example 6.36 INSERT using Defaults

- Insert a new row into Staff table supplying data for all mandatory columns.

```
Assignment Project Exam Help  
INSERT INTO Staff(staffNo, fName, lName,  
VALUES ('S  
          https://eduassistpro.github.io/  
          'Assistant', 810  
          'Add WeChat edu_assist_pro
```

- Or

```
INSERT INTO Staff  
VALUES ('SG44', 'Anne', 'Jones', 'Assistant', NULL,  
NULL, 8100, 'B003');
```

# INSERT ... SELECT

- Second form of INSERT allows multiple rows to be copied from one or more tables to another:

Assignment Project Exam Help

INSERT INTO <https://eduassistpro.github.io/> [list])  
SELECT ..

Add WeChat edu\_assist\_pro

## Example 6.37 INSERT ... SELECT (1 of 3)

Assume there is a table StaffPropCount that contains names of staff and number of properties they manage:

Assignment Project Exam Help

StaffPropC <https://eduassistpro.github.io/>

Add WeChat [edu\\_assist\\_pro](#)  
Populate StaffPropCount using propertyForRent  
tables.

## Example 6.37 INSERT ... SELECT (2 of 3)

```
INSERT INTO StaffPropCount  
  (SELECT s.staffNo, fName, lName, COUNT(*)  
   FROM Staff s, PropertyForRent p  
   WHERE s.staffNo = p.staffNo  
   GROUP BY s.staffNo)  
UNION  
  (SELECT staffNo, fName, lName  
   FROM Staff  
   WHERE staffNo NOT IN  
         (SELECT DISTINCT staffNo  
          FROM PropertyForRent));
```

## Example 6.37 INSERT ... SELECT (3 of 3)

staffNo	fName	lName	propCount
SG14	David	Ford	1
SL21	John	White	0
SG37			2
SA9			1
SG5	Susan	B	
SL41	Julie	L	

- If second part of UNION is omitted, excludes those staff who currently do not manage any properties.

# UPDATE (1 of 2)

UPDATE TableName

SET columnName1 = dataValue1

[columnName2 = dataValue2]

[WHERE sea

<https://eduassistpro.github.io/>

- **TableName** can be name of a table or an updatable view.
- SET clause specifies names of one or more columns that are to be updated.

## UPDATE (2 of 2)

- WHERE clause is optional:
  - if omitted, named columns are updated for all rows in table; **Assignment Project Exam Help**
  - If specified, **searchCon** <https://eduassistpro.github.io/tisfy>
- New **dataValue(s)** must be co **Add WeChat** **edu\_assist\_pro** with data type for corresponding column.

## Example 6.38/39 UPDATE All Rows

Give all staff a 3% pay increase.

UPDATE Staff

Assignment Project Exam Help

SET salary

<https://eduassistpro.github.io/>

Give all Manager

Add WeChat edu\_assist\_pro

UPDATE Staff

SET salary = salary\*1.05

WHERE position = 'Manager';

## Example 6.40 UPDATE Multiple Columns

Promote David Ford (staffNo='SG14') to Manager and change his salary to £18,000.

Assignment Project Exam Help

```
UPDATE S https://eduassistpro.github.io/
SET position = 'Manager',          000
WHERE staffNo = 'SG14';
      Add WeChat edu\_assist\_pro
```

# DELETE

DELETE FROM TableName  
[WHERE searchCondition]

Assignment Project Exam Help

- **TableName** can be a table or an updatable view. <https://eduassistpro.github.io/>
- **searchCondition** is optional; all rows are deleted from table. This does not delete table. If **search\_condition** is specified, only those rows that satisfy condition are deleted.

## Example 6.41/42 DELETE Specific Rows

Delete all viewings that relate to property PG4.

```
DELETE FROM Viewing  
          Assignment Project Exam Help  
WHERE pr  
          https://eduassistpro.github.io/
```

Delete all records from the Viewing  
 Add WeChat edu\_assist\_pro

```
DELETE FROM Viewing;
```

# Copyright

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro