P E S Education Trust (R), Mandya

P E S COLLEGE OF ENGINEERING

(An Autonomous Institution Affiliated to VTU, Belagavi, Grant -in- Aid Institution (Government of Karnataka), World Bank Funded College (TEQIP), Accredited by NBA & NAAC and Approved by AICTE, New Delhi.)

Mandya – 571401, Karnataka

Database Management System Lab

Department of Information Science and Engineering

P E S College of Engineering

Mandya – 571401, Karnataka

P E S Education Trust(R), Mandya

P E S College of Engineering

(An Autonomous Institution Affiliated to VTU, Belagavi)

Mandya - 571 401, Karnataka

Vision/ ಆಶಯ

“PESCE shall be a leading institution imparting quality engineering and management education developing creative and socially responsible professionals.”

"PESCE ಸೃಜನಶೀಲ ಮತ್ತು ಸಾಮಾಜಿಕ ಜವಾಬ್ದಾರಿಯುತ ವೃತ್ತಿಪರರನ್ನು ಅಭಿವೃದ್ಧಿಪಡಿಸುವ ಗುಣಮಟ್ಟದ ಎಂಜಿನಿಯರಿಂಗ್ ಮತ್ತು ನಿರ್ವಹಣಾ ಶಿಕ್ಷಣವನ್ನು ನೀಡುವ ಪ್ರಮುಖ ಸಂಸ್ಥೆಯಾಗಿದೆ."

Mission/ ಧ್ಯೇಯ

- Provide state of the art infrastructure, motivate the faculty to be proficient in their field of specialization and adopt best teaching-learning practices.

  ಅತ್ಯಾಧುನಿಕ ಮೂಲಸೌಕರ್ಯಗಳನ್ನು ಒದಗಿಸಿ, ಬೋಧಕವರ್ಗವನ್ನು ತಮ್ಮ ವಿಶೇಷ ಕ್ಷೇತ್ರದಲ್ಲಿ ಪ್ರವೀಣರಾಗುವಂತೆ ಪ್ರೇರೇಪಿಸಿ ಮತ್ತು ಅತ್ಯುತ್ತಮ ಬೋಧನೆ-ಕಲಿಕೆಯ ಅಭ್ಯಾಸಗಳನ್ನು ಅಳವಡಿಸಿಕೊಳ್ಳಿ.

- Impart engineering and managerial skills through competent and committed faculty using outcome based educational curriculum.

  ಫಲಿತಾಂಶ ಆಧಾರಿತ ಶೈಕ್ಷಣಿಕ ಪಠ್ಯಕ್ರಮವನ್ನು ಬಳಸಿಕೊಂಡು ಸಮರ್ಥ ಮತ್ತು ಬದ್ಧ ಅಧ್ಯಾಪಕರ ಮೂಲಕ ಎಂಜಿನಿಯರಿಂಗ್ ಮತ್ತು ನಿರ್ವಾಹಕ ಕೌಶಲ್ಯಗಳನ್ನು ನೀಡಿ.

- Inculcate professional ethics, leadership qualities and entrepreneurial skills to meet the societal needs.

  ಸಾಮಾಜಿಕ ಅಗತ್ಯಗಳನ್ನು ಪೂರೈಸಲು ವೃತ್ತಿಪರ ನೈತಿಕತೆ, ನಾಯಕತ್ವ ಗುಣಗಳು ಮತ್ತು ಉದ್ಯಮಶೀಲತೆಯ ಕೌಶಲ್ಯಗಳನ್ನು ರೂateಿಸಿಕೊಳ್ಳಿ.

- Promote research, product development and industry-institution interaction.

  ಸಂಶೋಧನೆ, ಉತ್ಪನ್ನ ಅಭಿವೃದ್ಧಿ ಮತ್ತು ಉದ್ಯಮ-ಸಂಸ್ಥೆಗಳ ಪರಸ್ಪರ ಕ್ರಿಯೆಯನ್ನು ಉತ್ತೇಜಿಸಿ.

P E S Education Trust(R), Mandya

P E S College of Engineering

(An Autonomous Institution Affiliated to VTU, Belagavi)

Department of Information Science & Engineering

About the Department/ ಇಲಾಖೆಯ ಬಗ್ಗೆ

The Department of Information science and Engineering takes pride in producing quality engineers over the past 20 years. The credit for all the flowery results goes to the highly motivating staff, from whom all students draw inspiration. The Department was started in the year 2000. The present intake of the undergraduate program is 60. The department has well equipped classrooms, computer laboratories with high-end systems, department library and good collection of software's. Also a research centre is a major credential to our department. We are proud to produce the first PhD student in our college. Faculty members of the department are involved in research activities in different fields such as Medical Image Processing, Pattern Recognition, and Data Mining etc. The department is using Outcome-based education (OBE), which is a recurring education reform model, and it is affiliated to Visvesvaraya Technological University (VTU). The department has achieved good Placement, conducted International /national Conferences and other sponsored short-term courses, workshops, National seminars and symposia. The laboratory facilities and the Internet access are available round the clock to the staff and students of the Information Science and Engineering.

ಮಾಹಿತಿ ವಿಜ್ಞಾನ ಮತ್ತು ಎಂಜಿನಿಯರಿಂಗ್ ವಿಭಾಗವು ಕಳೆದ 20 ವರ್ಷಗಳಲ್ಲಿ ಗುಣಮಟ್ಟದ ಎಂಜಿನಿಯರ್‌ಗಳನ್ನು ಉತ್ಪಾದಿಸುವಲ್ಲಿ ಹೆಮ್ಮೆ ಪಡುತ್ತದೆ. ಎಲ್ಲಾ ಹೂವುಗಳ ಫಲಿತಾಂಶಗಳ ಕ್ರೆಡಿಟ್ ಹೆಚ್ಚು ಪ್ರೇರೇಪಿಸುವ ಸಿಬ್ಬಂದಿಗೆ ಸಲ್ಲುತ್ತದೆ, ಅವರಿಂದ ಎಲ್ಲಾ ವಿದ್ಯಾರ್ಥಿಗಳು ಸ್ಫೂರ್ತಿ ಪಡೆಯುತ್ತಾರೆ. ಇಲಾಖೆಯು 2000 ನೇ ವರ್ಷದಲ್ಲಿ ಆರಂಭವಾಯಿತು. ಪದವಿಪೂರ್ವ ಕಾರ್ಯಕ್ರಮದ ಪ್ರಸ್ತುತ ಸೇವನೆಯು 60. ಇಲಾಖೆಯು ಸುಸಜ್ಜಿತವಾದ ತರಗತಿ ಕೊಠಡಿಗಳು, ಉನ್ನತ ಮಟ್ಟದ ವ್ಯವಸ್ಥೆಗಳೊಂದಿಗೆ ಕಂಪ್ಯೂಟರ್ ಪ್ರಯೋಗಾಲಯಗಳು, ಇಲಾಖೆಯ ಗ್ರಂಥಾಲಯ ಮತ್ತು ಸಾಫ್ಟ್‌ವೇರ್‌ಗಳ ಉತ್ತಮ ಸಂಗ್ರಹವನ್ನು ಹೊಂದಿದೆ. ಅಲ್ಲದೆ ಒಂದು ಸಂಶೋಧನಾ ಕೇಂದ್ರವು ನಮ್ಮ ಇಲಾಖೆಗೆ ಪ್ರಮುಖ ರುಜುವಾತು. ನಮ್ಮ ಕಾಲೇಜಿನಲ್ಲಿ ಮೊದಲ ಪಿಎಚ್‌ಡಿ ವಿದ್ಯಾರ್ಥಿಯನ್ನು ತಯಾರಿಸಲು ನಮಗೆ ಹೆಮ್ಮೆ ಇದೆ. ಇಲಾಖೆಯ ಅಧ್ಯಾಪಕರು ವೈದ್ಯಕೀಯ ಚಿತ್ರ ಸಂಸ್ಕರಣೆ, ಪ್ಯಾಟರ್ನ್ ರೆಕಗ್ನಿಷನ್, ಮತ್ತು ಡೇಟಾ ಮೈನಿಂಗ್ ಮುಂತಾದ ವಿವಿಧ ಕ್ಷೇತ್ರಗಳಲ್ಲಿ ಸಂಶೋಧನಾ ಚಟುವಟಿಕೆಗಳಲ್ಲಿ ತೊಡಗಿಸಿಕೊಂಡಿದ್ದಾರೆ. ಇಲಾಖೆಯು ಫಲಿತಾಂಶ ಆಧಾರಿತ ಶಿಕ್ಷಣವನ್ನು (ಒಬಿಇ) ಬಳಸುತ್ತಿದೆ, ಇದು ಪುನರಾವರ್ತಿತ ಶಿಕ್ಷಣ ಸುಧಾರಣಾ ಮಾದರಿಯಾಗಿದೆ, ಮತ್ತು ಇದು ವಿಶ್ವೇಶ್ವರಯ್ಯ ತಾಂತ್ರಿಕ ವಿಶ್ವವಿದ್ಯಾಲಯಕ್ಕೆ (ವಿಟಿಯು) ಸಂಯೋಜಿತವಾಗಿದೆ. ಇಲಾಖೆಯು ಉತ್ತಮ ಉದ್ಯೋಗವನ್ನು ಸಾಧಿಸಿದೆ, ಅಂತರಾಷ್ಟ್ರೀಯ /ರಾಷ್ಟ್ರೀಯ ಸಮ್ಮೇಳನಗಳನ್ನು ಮತ್ತು ಇತರ ಪ್ರಾಯೋಜಿತ ಅಲ್ಪಾವಧಿ ಕೋರ್ಸ್‌ಗಳು, ಕಾರ್ಯಾಗಾರಗಳು, ರಾಷ್ಟ್ರೀಯ ವಿಚಾರಗೋಷ್ಠಿಗಳು ಮತ್ತು ವಿಚಾರ ಸಂಕಿರಣಗಳನ್ನು ನಡೆಸಿದೆ. ಪ್ರಯೋಗಾಲಯದ ಸೌಲಭ್ಯಗಳು ಮತ್ತು ಇಂಟರ್ನೆಟ್ ಪ್ರವೇಶವು ಸಿಬ್ಬಂದಿ ಮತ್ತು ಮಾಹಿತಿ ವಿಜ್ಞಾನ ಮತ್ತು ಎಂಜಿನಿಯರಿಂಗ್‌ನ ವಿದ್ಯಾರ್ಥಿಗಳಿಗೆ 24 ಗಂಟೆಯೂ ಲಭ್ಯವಿದೆ.

P E S Education Trust(R), Mandya

P E S College of Engineering

(An Autonomous Institution Affiliated to VTU, Belagavi)

Department of Information Science & Engineering

Vision/ ಆಶಯ

“The department strives to equip our graduates with Knowledge and Skills to contribute significantly to Information Science & Engineering and enhance quality research for the benefit of society”.

"ಮಾಹಿತಿ ವಿಜ್ಞಾನ ಮತ್ತು ಎಂಜಿನಿಯರಿಂಗ್‌ಗೆ ಗಣನೀಯ ಕೊಡುಗೆ ನೀಡಲು ಮತ್ತು ಸಮಾಜದ ಪ್ರಯೋಜನಕ್ಕಾಗಿ ಗುಣಮಟ್ಟದ ಸಂಶೋಧನೆಯನ್ನು ಹೆಚ್ಚಿಸಲು ನಮ್ಮ ಪದವೀಧರರನ್ನು ಜ್ಞಾನ ಮತ್ತು ಕೌಶಲ್ಯಗಳೊಂದಿಗೆ ಸಜ್ಜುಗೊಳಿಸಲು ಇಲಾಖೆಯು ಶ್ರಮಿಸುತ್ತದೆ."

Mission/ ಧ್ಯೇಯ

- To provide students with state of art facilities and tools of Information Science & Engineering to become productive, global citizens and life-long learners.
  ವಿದ್ಯಾರ್ಥಿಗಳಿಗೆ ಅತ್ಯಾಧುನಿಕ ಸೌಲಭ್ಯಗಳು ಮತ್ತು ಮಾಹಿತಿ ವಿಜ್ಞಾನ ಮತ್ತು ಎಂಜಿನಿಯರಿಂಗ್ ಉಪಕರಣಗಳನ್ನು ಉತ್ಪಾದಕ, ಜಾಗತಿಕ ನಾಗರಿಕರು ಮತ್ತು ಜೀವನಪರ್ಯಂತ ಕಲಿಯುವವರನ್ನಾಗಿ ಮಾಡಲು .
- To prepare students for careers in IT industry, Higher education and Research.
  ಐಟಿ ಉದ್ಯಮ, ಉನ್ನತ ಶಿಕ್ಷಣ ಮತ್ತು ಸಂಶೋಧನೆಗಾಗಿ ವಿದ್ಯಾರ್ಥಿಗಳನ್ನು ತಯಾರಿಸಲು .
- To inculcate leadership qualities among students to make them competent Information Science & Engineering professionals or entrepreneurs.
  ವಿದ್ಯಾರ್ಥಿಗಳಲ್ಲಿ ನಾಯಕತ್ವ ಗುಣಗಳನ್ನು ಬೆಳೆಸಲು ಅವರನ್ನು ಸಮರ್ಥ ಮಾಹಿತಿ ವಿಜ್ಞಾನ ಮತ್ತು ಎಂಜಿನಿಯರಿಂಗ್ ವೃತ್ತಿಪರರು ಅಥವಾ ಉದ್ಯಮಿಗಳನ್ನಾಗಿ ಮಾಡಲು .

Program Educational Objectives (PEOs): Graduates of the program will be able to

PEO1: Establish a productive Information Science & Engineering career in industry, government or academia.

PEO2: Interact with their peers in other disciplines by exhibiting professionalism and team work to contribute to the economic growth of the country.

PEO3: Promote the development of innovative systems and solutions to the problems in Information Science using hardware and software integration.

PEO4: Pursue higher studies in Engineering, Management or Research.

Program Specific Outcomes (PSOs)

PSO1. Analyze, design, develop and test the principles of System software and Database concepts for computer-based systems.

PSO2. Develop computer communication systems and applications for Information security.

PSO3. Apply the knowledge of Information Science and Engineering to solve any software and hardware related problems and to organize, manage and monitor IT Infrastructure.

Program Outcomes (POs)

PO1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Course Overview

A database management system (DBMS) is computer application software that provides a way to manage data. The requirement of modern days is to have an automated system that manages, modifies, and updates data accurately. This is achieved by a DBMS in robust, correct, and non-redundant way. Structured Database Management Systems (DBMS) based on relational and other models have long formed the basis for such databases. Consequently, Oracle, Microsoft SQL Server, Sybase etc. have emerged as leading commercial systems while MySQL, PostgreSQL etc. lead in open source and free domain. The Course allows students to apply the conceptual design model to construct the real-world requirement. Course gives familiarity of Database Concepts were students can analyze the various constraints to populate the database and examine different working concepts of DBMS to infer the most suitable pattern of documentation.

DBMS lab with mini project aims at practicing and achieving this aim by using MySQL. While also gain capability to design database and its hierarchical structure for given real world application.

Course Objectives

The objectives of this course are to make students to learn,

1. To understand the foundation knowledge in database concepts, technology and practice to prepare students into well-informed database application developers.
2. Strong practice in SQL programming through a variety of database problems.
3. Develop database applications using front-end tools and back-end DBMS.

Course Outcomes

After learning all the units of the course, students is able to:

1. Understand database language commands to create simple database
2. Analyze the database using queries to retrieve records
3. Analyze front end tools to design forms, reports and menus
4. Develop solutions using database concepts for real time requirements.

P E S Education Trust(R), Mandya

P E S College of Engineering

(An Autonomous Institution Affiliated to VTU, Belagavi)

Department of Information Science & Engineering

Syllabus

PART-A: SQL Programming

Note:

- Design, develop, and implement the specified queries for the following problems using Oracle, MySQL, MS SQL Server, or any other DBMS under LINUX/Windows environment.
- Create Schema and insert at least 5 records for each table. Add appropriate database constraints.

1. Implementation of DDL commands of SQL with suitable examples

   a) Create database

   b) Create table

   c) Alter table

   d) Drop Table

2. Study and Implementation of different types of constraints.

3. Implementation of DML commands of SQL with suitable examples

   a) Insert

   b) Update

   c) Delete

   d) Alter

4. Implementation of different types of operators in SQL

   a) Arithmetic Operators

   b) Logical Operators

   c) Comparison Operator

   d) Special Operator

   e) Set Operation

5. Implementation of different types of function with suitable examples

   a) Number function

   b) Aggregate Function

   c) Character Function

   d) Conversion Function

   e) Date Function

6. Study and Implementation of
   - a) Group By & having clause
   - b) Order by clause
   - c) Nested queries
   - d) Views
7. Implementation of different types of Joins
   - a) Inner Join
   - b) Outer Join
   - c) Natural Join etc.
8. Study and implementation of Database Backup and Recovery commands.
9. Study and implementation of Rollback, Commit, Save-point.

PART – B Mini Project

Develop a menu driven project for of database management system

- ➢ For any problem selected
- ➢ Make sure that the application should have five or more tables
- ➢ Use Java, C#, Python, or any other similar front-end tool.
- ➢ All applications must be demonstrated on desktop/laptop as a stand-alone. or web based application (Mobile apps on Android/IOS are not permitted.)
- ➢ Indicative areas include; Health care, Education, Transportation

Introduction to Database Management System

Today, the success of any organization depends on its ability to acquire accurate and timely data about its operations, to manage this data effectively, and to use this data to analyze and guide its activities. Phrases such as the information super highway have become ubiquitous, and information processing is a rapidly growing industry.

The amount of information available to us is increasing at an exploding rate, and the value of data as an organizational asset is widely recognized. Yet without the ability to manage this vast amount of data, and to quickly find the information that is relevant to a given scenario or interest, as the amount of information increases, it tends to become a distraction and a liability, rather than an asset. This paradox drives the need for increasingly powerful and flexible data management systems. To get the most out of their large and complex datasets, users must have tools that simplify the tasks of managing the data and extracting useful information in a timely fashion. Otherwise, data can become a liability, with the cost of acquiring it and managing it far exceeding the value that is derived from it.

Database: A database is a collection of related data, typically describing the activities of one or more related organizations.

For example: A university database might contain information about the following:

Entities are students, faculty and courses And **Relationships** between entities, such as students' enrollment in courses, and faculty teaching courses.

Database Management System (DBMS) : A database management system (DBMS) is software that enables users to create and maintain a database. The DBMS is hence a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications.

Applications of Database Systems:

Databases are widely used. Some applications of database systems are given below:

1. **Banking:** For customer information, accounts, and loans, and banking transactions.

2. **Airlines:** For reservations and schedule information.
3. **Universities:** For student information, course registrations, and grades.
4. **Credit card transactions:** For purchases on credit cards and generation of monthly statements.
5. **Telecommunication:** For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.
6. **Finance:** For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds.
7. **Sales:** For customer, product, and purchase information.
8. **Manufacturing:** For management of supply chain and for tracking production of items in factories, inventories of items in warehouses/stores, and orders for items.
9. **Human resources:** For information about employees, salaries, payroll taxes and benefits, and for generation of paychecks.

Advantages of DBMS:

Using a DBMS to manage data has following advantages:

1. **Data independence:** Application programs should be as independent as possible from details of data representation and storage. The DBMS can provide an abstract view of the data to insulate application code from such details.
2. **Efficient data access:** A DBMS utilizes a variety of sophisticated techniques to store and retrieve data efficiently. This feature is especially important if the data is stored on external storage devices.
3. **Data integrity and security:** If data is always accessed through the DBMS, the DBMS can enforce integrity constraints on the data. For example, before inserting salary information for an employee, the DBMS can check that the department budget is not exceeded. Also, the DBMS can enforce *access controls* that govern what data is visible to different classes of users.
4. **Data administration:** When several users share the data, centralizing the administration of data can offer significant improvements. Experienced professionals, who understand the nature of the data being managed, and how different groups of users use it, can be responsible for organizing the data representation to minimize redundancy and for fine-tuning the storage of the data to make retrieval efficient.

5. **Concurrent access and crash recovery:** A DBMS schedules concurrent accesses to the data in such a manner that users can think of the data as being accessed by only one user at a time. Further, the DBMS protects users from the effects of system failures.
6. **Reduced application development time:** Clearly, the DBMS supports many important functions that are common to many applications accessing data stored in the DBMS. This, in conjunction with the high-level interface to the data, facilitates quick development of applications. Such applications are also likely to be more robust than applications developed from scratch because many important tasks are handled by the DBMS instead of being implemented by the application.

People dealing with databases:

The categories of people who deal with databases:

1. **Database Implementers:** People in this category are associated with the creation and use of databases. These people build DBMS software. Database implementers work for vendors such as IBM or Oracle.
2. **End Users:** End users store and use data in a DBMS. End users come from a diverse and increasing number of fields. Many end users simply use applications written by database application programmers, and so require little technical knowledge about DBMS software. However, sophisticated users who make more extensive use of a DBMS, such as writing their own queries, require a deeper understanding of its features.
3. **Database application programmers:** These people develop packages that facilitate data access for end users using the host or data languages and software tools that DBMS vendors provide. (Such tools include report writers, spreadsheets, etc.)

Database administrator (DBA): The DBA is responsible for authorizing access to the database, for coordinating and monitoring its use, and for acquiring software and hardware resources as needed. The DBA is accountable for problems such as breach of security or poor system response time.

ENTITY-RELATIONSHIP MODEL

A database can be modeled as, a collection of entities and Relationships among entities.

Entity: A real-world object that can be distinctly identified may represent some real physical object. May represent some conceptual idea **Example:** specific person, company, event, plant

Entity set: An entity set is a set of entities of the same type that share the same properties. **Example:** set of all persons, companies, trees, holidays.

An Entity should be, An Object that will have many instances in the Database, An Object that will have multiple attributes and An Object that we are trying to model. An Entity should not be, User of the Database and An output of the Database (ex. Report).

Attributes: An entity is represented by a set of attributes, that is, descriptive properties possessed by all members of an entity set. (value from corresponding entity)

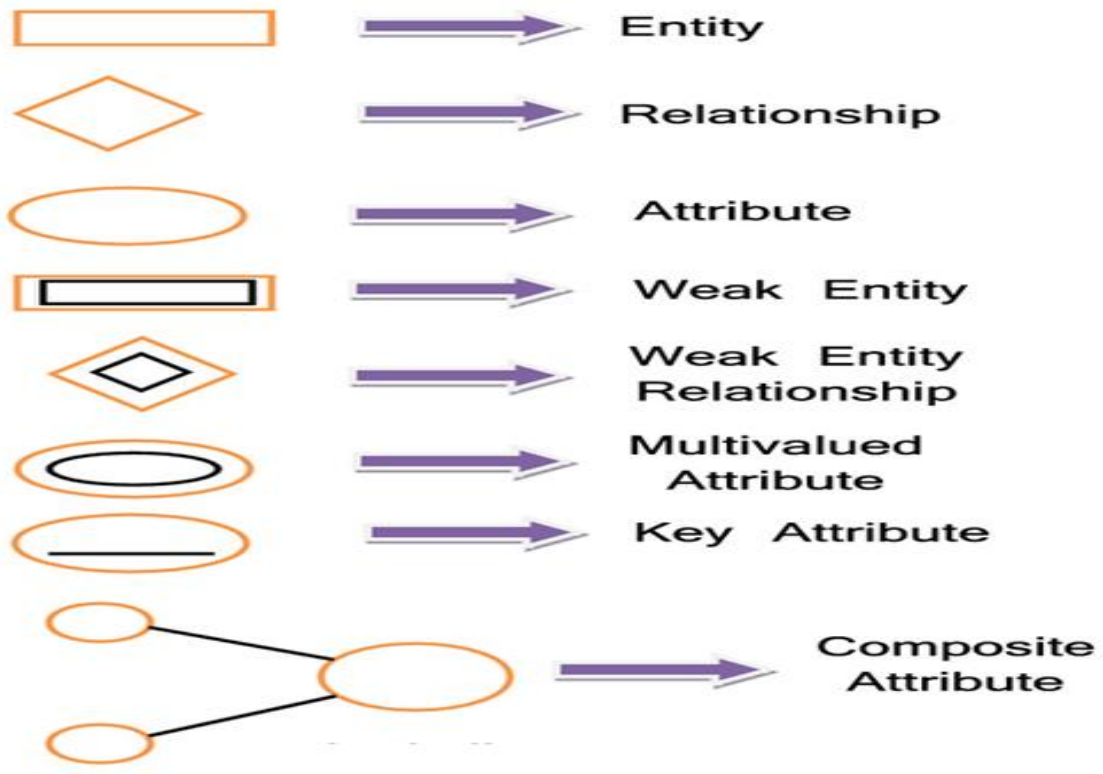
Example: customer = (customer-name, social-security, customer-street, customer-city)

account = (account-number, balance)

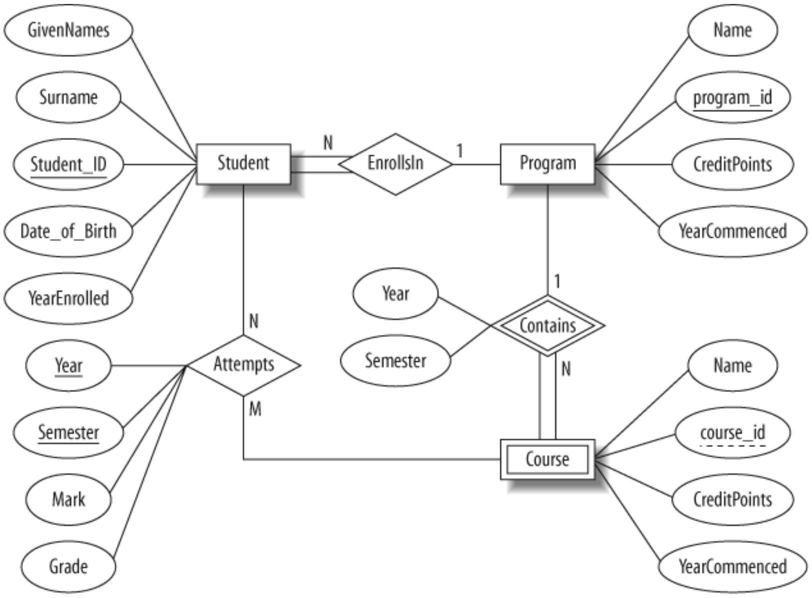
Domain: Domain is the set of permitted values for each attribute.

Attribute types: Simple and composite attributes, Single-valued and multi-valued attributes, Null attributes, Derived attributes and Identifiers (Key) attributes etc.

Symbols Used in ER Diagram:



Example of ER Diagram:



Introduction to SQL

SQL stands for Structured Query Language. Oracle provides many extensions to ANSI SQL. SQL is standard language for interacting with a relational database. A table is primary database object of SQL, which is used to store the data in the form of rows and columns. SQL developed by IBM is used as standard language to access data from database. In order to communicate with database SQL has been divided into four sub languages and each sub language consists of different commands, they are:

Data Definition Language (DDL): DDL commands are used to define the database structure or schema. Different DDL commands are:

- **CREATE** - to create objects in the database
- **ALTER** - alters the structure of the database
- **DROP** - delete objects from the database
- **TRUNCATE** - remove all records from a table, including all spaces allocated for the records are removed
- **RENAME** - rename an object

Data Manipulation Language (DML): DML commands are used for managing data within schema objects. Different DML commands are:

- **SELECT** - retrieve data from the a database
- **INSERT** - insert data into a table
- **UPDATE** - updates existing data within a table
- **DELETE** - deletes all records from a table, the space for the records remain

Data Control Language (DCL): DCL commands are used to control the access to the database objects. Different DCL commands are:

- **GRANT** - gives user's access privileges to database
- **REVOKE** - withdraw access privileges given with the GRANT command

Transaction Control Language (TCL): TCL commands are used to manage the changes made by DML statements. It allows statements to be grouped together into logical transactions. Different TCL commands are:

- **COMMIT** - save work done
- **SAVEPOINT** - identify a point in a transaction to which you can later roll back
- **ROLLBACK** - restore database to original since the last COMMIT

CREATE TABLE Statement:

The CREATE TABLE Statement is used to create tables to store data. Integrity Constraints like primary key, unique key and foreign key can be defined for the columns while creating the table. The integrity constraints can be defined at column level or table level. The implementation and the syntax of the CREATE Statements differs for different RDBMS.

The Syntax for the CREATE TABLE Statement is:

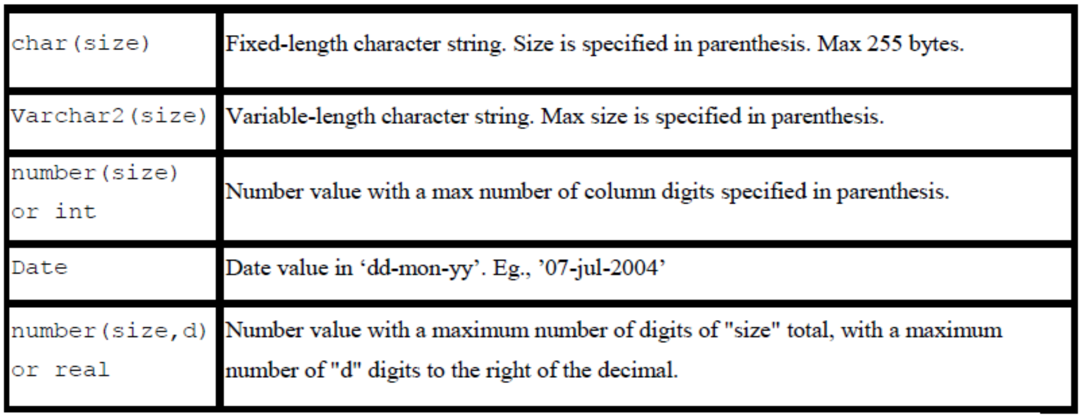
```
CREATE TABLE table_name
                    (column_name1 datatype constraint,
                     column_name2 datatype, ...
                     column_nameNdatatype);
```

table_name - is the name of the table.

column_name1, column_name2.... - is the name of the columns

datatype - is the datatype for the column like char, date, number etc.

SQL Data Types:



char(size)	Fixed-length character string. Size is specified in parenthesis. Max 255 bytes.
Varchar2(size)	Variable-length character string. Max size is specified in parenthesis.
number(size) or int	Number value with a max number of column digits specified in parenthesis.
Date	Date value in ‘dd-mon-yy’. Eg., ’07-jul-2004’
number(size,d) or real	Number value with a maximum number of digits of "size" total, with a maximum number of "d" digits to the right of the decimal.

SQL Integrity Constraints:

Integrity Constraints are used to apply business rules for the database tables. The constraints available in SQL are Foreign Key, Primary key, Not Null, Unique and Check.

Constraints can be defined in two ways:

1. The constraints can be specified immediately after the column definition. This is called column-level definition.
2. The constraints can be specified after all the columns are defined. This is called table level definition.

1) Primary key:

This constraint defines a column or combination of columns which uniquely identifies each row in the table.

Syntax to define a Primary key at column level:

Column_namedatatype [CONSTRAINT constraint_name] PRIMARY KEY

Syntax to define a Primary key at table level:

[CONSTRAINT constraint_name] PRIMARY KEY (column_name1,
column_name2,..)

column_name1, column_name2 are the names of the columns which define the primary key.

The syntax within the bracket i.e. [CONSTRAINT constraint_name] is optional.

2) Foreign key or Referential Integrity:

This constraint identifies any column referencing the PRIMARY KEY in another table. It establishes a relationship between two columns in the same table or between different tables. For a column to be defined as a Foreign Key, it should be a defined as a Primary Key in the table which it is referring. One or more columns can be defined as foreign key.

Syntax to define a foreign key at column level:

[constraint constraint_name] references referenced_table_name(column_name)

Syntax to define a Foreign key at table level:

[constraint constraint_name] foreign key(column_name) references
referenced_table_name(column_name);

3) Not Null Constraint:

This constraint ensures all rows in the table contain a definite value for the column which is specified as not null. Which means a null value is not allowed.

Syntax to define a Not Null constraint:

[CONSTRAINT constraint name] NOT NULL

4) Unique Key:

This constraint ensures that a column or a group of columns in each row have a distinct value. A column(s) can have a null value but the values cannot be duplicated.

Syntax to define a Unique key at column level:

[CONSTRAINT constraint_name] UNIQUE

Syntax to define a Unique key at table level:

[CONSTRAINT constraint_name] UNIQUE(column_name)

5) Check Constraint:

This constraint defines a business rule on a column. All the rows must satisfy this rule. The constraint can be applied for a single column or a group of columns.

Syntax to define a Check constraint:

[CONSTRAINT constraint_name] CHECK (condition)

ALTER TABLE Statement

The SQL ALTER TABLE command is used to modify the definition structure) of a table by modifying the definition of its columns. The ALTER command is used to perform the following functions.

1) Add, drop, modify table columns
2) Add and drop constraints
3) Enable and Disable constraints

Syntax to add a column

ALTER TABLE table_name ADD column_namedatatype;

For Example: To add a column "experience" to the employee table, the query would be like

ALTER TABLE employee ADD experience number(3);

Syntax to drop a column

ALTER TABLE table_name DROP column_name;

For Example: To drop the column "location" from the employee table, the query would be like

ALTER TABLE employee DROP location;

Syntax to modify a column

ALTER TABLE table_name MODIFY column_namedatatype;

For Example: To modify the column salary in the employee table, the query would be like

ALTER TABLE employee MODIFY salary number(15,2);

Syntax to add PRIMARY KEY constraint

ALTER TABLE table_nameADD CONSTRAINT constraint_name PRIMARY KEY column_name;

Syntax to drop PRIMARY KEY constraint

ALTER TABLE table_nameDROP PRIMARY KEY;

The DROP TABLE Statement

The DROP TABLE statement is used to delete a table.

Syntax: DROP TABLE table_name;

TRUNCATE TABLE Statement

What if we only want to delete the data inside the table, and not the table itself? Then, use the TRUNCATE TABLE statement:

Syntax: TRUNCATE TABLE table_name;

Data Manipulation Language (DML):

The SELECT Statement

The SELECT statement is used to select data from a database. The result is stored in a result table, called the result-set.

SELECT Syntax:

SELECT * FROM table_name;

The SELECT DISTINCT Statement

In a table, some of the columns may contain duplicate values. This is not a problem, however, sometimes you will want to list only the different (distinct) values in a table. The DISTINCT keyword can be used to return only distinct (different) values.

SELECT DISTINCT Syntax:

SELECT DISTINCT column_name(s)
FROM table_name;

The WHERE Clause

The WHERE clause is used to extract only those records that fulfill a specified criterion.

WHERE Syntax:

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value;
```

The AND & OR Operators

The **AND** operator displays a record if both the first condition and the second condition is true.

The **OR** operator displays a record if either the first condition or the second condition is true.

The ORDER BY Clause

The ORDER BY clause is used to sort the result-set by a specified column.

The ORDER BY clausesort the records in ascending order by default.

If you want to sort the records in a descending order, you can use the DESC keyword.

ORDER BY Syntax:

```
SELECT column_name(s)
FROM table_name
ORDER BY column_name(s) ASC|DESC;
```

The GROUP BY Clause

The GROUP BY clause can be used to create groups of rows in a table. Group functions can be applied on such groups.

GROUP BY Syntax;

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value
GROUP BY column_name(s);
```

Aggregate Functions:

Aggregate functions perform a calculation on a set of values and return a single value. It means that, the data that you need is not always stored in the tables. However, you can get it by performing the calculations of the stored data when you select it.

Aggregate Functions are all about

- Performing calculations on multiple rows
- Of a single column of a table
- And returning a single value.

The ISO standard defines five (5) aggregate functions namely;

1. COUNT
2. SUM
3. AVG
4. MIN
5. MAX

Except for COUNT, all other aggregate functions ignore null values. Aggregate functions are frequently used with the GROUP BY clause of the SELECT statement.

The HAVING clause:

The HAVING clause can be used to restrict the display of grouped rows. The result of the grouped query is passed on to the HAVING clause for output filtration.

HAVING Syntax;

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value
GROUP BY column_name(s)
HAVING condition;
```

The INSERT INTO Statement

The INSERT INTO statement is used to insert a new row in a table.

SQL INSERT INTO Syntax:

It is possible to write the INSERT INTO statement in two forms.

The first form doesn't specify the column names where the data will be inserted, only their values:

```
INSERT INTO table_nameVALUES (value1, value2, value3,...);
OR
INSERT INTO table_nameVALUES(&column1, &column2, &column3,...);
```

The second form specifies both the column names and the values to be inserted:

```
INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...);
```

The UPDATE Statement:

The UPDATE statement is used to update existing records in a table.

SQL UPDATE Syntax:

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value;
```

The DELETE Statement

The DELETE statement is used to delete rows in a table.

SQL DELETE Syntax:

```
DELETE FROM table_name
WHERE some_column=some_value;
```

Transaction Control language

Transaction Control Language (TCL) commands are used to manage transactions in database. These are used to manage the changes made by DML statements. It also allows statements to be grouped together into logical transactions

Commit command

Commit command is used to permanently save any transaction into database.

Commit command's syntax: **commit;**

Rollback command

This command restores the database to last commited state. It is also use with savepoint command to jump to a savepoint in a transaction.

Rollback command's syntax: **rollback to savepoint_name;**

Savepoint command

Savepoint command is used to temporarily save a transaction so that you can rollback to that point whenever necessary.

Savepoint command's syntax: **savepoint savepoint_name;**

Data Control Language

Data Control Language(DCL) is used to control privilege in Database. To perform any operation in the database, such as for creating tables, sequences or views we need privileges.

Privileges are of two types,

System: creating session, table etc are all types of system privilege.

Object: any command or query to work on tables comes under object privilege.

DCL defines two commands,

Grant: Gives user access privileges to database.

Revoke: Take back permissions from user.

Lab Experiments

Consider the **Insurance** database given below. The primary keys are underlined and the data types are specified:

PERSON (driver-id:string,name:string,address:string)

CAR (Regno:string,model:string,year:int)

ACCIDINT (report-number:int,date:date,location:string)

OWNS (driver-id:string,regno:string)

PARTICIPATED (driver-id:string,regno:string,report-number:int,damage-amount: int)

1) create the above tables by properly specifying the primary keys and the foreign keys
2) Enter atleast five tuples for each relation
3) Demonstrate how you
   a) update the damage amount for the car with a specific regno in accident with report number 12 to 25000
   b) add a new accident to the database
4) Find the total number of people who owned cars that were involved in accidents in 2002.
5) Find the number of accidents in which cars belonging to a specific model were involved.

Table creation:

```
SQL> create table Person (
   Driverid varchar(15),
   Name varchar(15) not null,
   Address varchar(20),
   primary key (Driverid));
Table created.
```

```
SQL> create table Car(
   Regno varchar(9),
   Model varchar(15) not null,
   Year integer not null,
   primary key (Regno));
Table created.
```

```
SQL> create table Accident(
   Reportno integer,
   Accdate date not null,
   Location varchar(15) not null,
   primary key (Reportno));
Table created.

SQL> create table Owns(
   Driverid varchar(15),
   Regno varchar(9),
   primary key (Driverid,Regno),
   foreign key (Driverid) references Person(Driverid),
   foreign key (Regno) references Car(Regno));
Table created.

SQL> create table Participated(
   Driverid varchar(15),
   Regno varchar(9),
   Reportno integer,
   Damageamount integer,
   primary key (Driverid,Regno,Reportno),
   foreign key (Driverid) references Person(Driverid),
   foreign key (Regno) references Car(Regno),
   foreign key (Reportno) references Accident(Reportno));
Table created.

SQL> desc Person;

 Name                         Null?                      Type
 ------------------------     --------                   --------------------------
 DRIVERID                     NOT NULL                    VARCHAR2 (15)
 NAME                          NOT NULL                    VARCHAR2 (15)
 ADDRESS                                                    VARCHAR2 (20)
SQL> insert into Person values ('1111','Ramu','Jayanagar');

1 row created.

SQL> insert into Person values ('2222','Manu','Rajajinagar');

1 row created.

SQL> insert into Person values ('3333','Pandit','Indiranagar');

1 row created.

SQL> insert into Person values ('4444','Gopal','BTMLayout');

1 row created.
```

```
SQL> insert into Person values ('5555','Lalit','Whitefield');

1 row created.

SQL> select * from Person;

DRIVERID        NAME              ADDRESS
---------------  ---------------   --------------------
1111              Ramu               Jayanagar
2222              Manu               Rajajinagar
3333             Pandit              Indiranagar
4444             Gopal              BTM Layout
5555              Lalit              Whitefield
SQL> desc Car;

 Name                                   Null?           Type
 --------------------------------        --------       ----------------------
 REGNO                                 NOT NULL         VARCHAR2 (9)
 MODEL                                 NOT NULL        VARCHAR2 (15)
 YEAR                                  NOT NULL        NUMBER (38)
SQL> insert into Car values ('KA04Q2301','Maruthi', 2000);

1 row created.

SQL> insert into Car values ('KA05P1000','Ford', 2002);

1 row created.

SQL> insert into Car values ('KA03L1234','Honda', 1999);

1 row created.

SQL> insert into Car values ('KA03L9999','Tata',2002);

1 row created.

SQL> insert into Car values('KA01P4026','Toyota',2003);

1 row created.

SQL> select * from Car;

REGNO                            MODEL                                  YEAR
---------                      ---------------                        --------
KA04Q2301                          Maruthi                              2000
KA05P1000                           Ford                                2002
KA03L1234                          Honda                                1999
KA03L9999                           Tata                                 2002
KA01P4026                         Toyota                                2003
```

```
SQL> desc Owns;
 Name                                   Null?                         Type
 ----------------------                --------                     ----------------
 DRIVERID                              NOT NULL                      VARCHAR2 (15)
 REGNO                                NOT NULL                        VARCHAR2 (9)
SQL> insert into Owns values ('1111','KA04Q2301');

1 row created.

SQL> insert into Owns values ('2222','KA05P1000');

1 row created.

SQL> insert into Owns values ('3333','KA03L1234');

1 row created.

SQL> insert into Owns values ('4444','KA03L9999');

1 row created.

SQL> insert into Owns values('5555','KA01P4026');

1 row created.

SQL> select * from Owns;

DRIVERID                             REGNO
---------------                        ---------
1111                                 KA04Q2301
2222                                 KA05P1000
3333                                 KA03L1234
4444                                 KA03L9999
5555                                 KA01P4026
SQL> desc Accident;

 Name                              Null?                           Type
 -----------------------------     --------                  ----------------------------
 REPORTNO                          NOT NULL                      NUMBER (38)
 ACCDATE                            NOT NULL                         DATE
 LOCATION                            NOT NULL                    VARCHAR2 (15)
SQL> insert into Accident values(12,'01-Jun-2001','Jayanagar');

1 row created.

SQL> insert into Accident values(25,'02-Jul-2002','AvenueRoad');

1 row created.

SQL> insert into Accident values (512,'08-Mar-2000','MGRoad');

1 row created.
```

```
SQL> insert into Accident values (1024,'25-Oct-2002','BrigadeRoad');

1 row created.

SQL> insert into Accident values (1000,'23-Dec-2003','RichmondCircle');

1 row created.

SQL> select * from Accident;

  REPORTNO              ACCDATE                     LOCATION
     ----------            ---------                 ----------------
      12                  01-JUN-01                        Jayanagar
      25                 02-JUL-02                      Avenue Road
     512                 08-MAR-00                       MG Road
     1024                25-OCT-02                      Brigade Road
     1000               23-DEC-03                     Richmond Circle
SQL> desc participated;
 Name                        Null?                        Type
 ---------------------------  --------          ---------------------------
 DRIVERID                      NOT NULL                VARCHAR2 (15)
 REGNO                          NOT NULL                 VARCHAR2 (9)
 REPORTNO                       NOT NULL                  NUMBER (38)
 DAMAGEAM                                                 NUMBER (38)
SQL> insert into Participated values ('1111','KA04Q2301', 12, 2000);

1 row created.

SQL> insert into Participated values ('2222','KA05P1000', 25, 15000);

1 row created.

SQL> insert into Participated values ('3333','KA03L1234', 512, 15500);

1 row created.

SQL> insert into Participated values ('4444','KA03L9999', 1024, 20000);

1 row created.

SQL> insert into Participated values ('5555','KA01P4026', 1000, 5000);

1 row created.

SQL> select * from Participated;

DRIVERID           REGNO            REPORTNO    DAMAGEAMOUNT

----------------    ---------          ----------    ------------

1111             KA04Q2301             12            2000
2222             KA05P1000              25            15000
3333             KA03L1234            512           15500
4444             KA03L9999           1024            20000
5555              KA01P4026          1000           5000
```

***** 1st QUERY *****

BEFORE:

```
SQL> select * from Participated;

DRIVERID          REGNO             REPORTNO          AMAGEAMOUNT
---------------   ----------        ----------          ------------
1111               KA04Q2301           12                   2000
2222               KA05P1000           25                    15000
3333               KA03L1234           512                   15500
4444               KA03L9999             1024                 20000
5555                KA01P4026            1000                  5000
SQL> update Participated
        set Damageamount=25000
        where Regno='KA04Q2301' and Reportno=12;

1 row updated.
```

AFTER:

```
SQL> select * from Participated;

DRIVERID          REGNO              REPORTNO      DAMAGEAMOUNT
---------------    ---------          ----------        ------------
1111               KA04Q2301         12                25000
2222                KA05P1000         25                 15000
3333               KA03L1234        512                   15500
4444                KA03L9999        1024                 20000
5555                 KA01P4026         1000                  5000
```

***** 2ND QUERY *****

```
SQL> insert into Person values ('6666','Bunty','Jayanagar');

1 row created.

SQL> select * from person;

DRIVERID           NAME                                   ADDRESS
--------------     --------------                      ------------------
1111                          Ramu                         Jayanagar
2222                          Manu                       Rajajinagar
3333                          Pandit                     Indiranagar
4444                          Gopal                    BTM Layout
5555                            Lalit                     Whitefield
6666                           Bunty                        Jayanagar

6 rows selected.
```

```
SQL> insert into Car values('KA052005','BMW',2005);

1 row created.

SQL> select * from car;

REGNO                     MODEL                        YEAR
---------               --------------               ----------
KA04Q2301               Maruthi                      2000
KA05P1000                Ford                        2002
KA03L1234                Honda                       1999
KA03L9999                 Tata                       2002
KA01P4026                Toyota                      2003
KA052005                  BMW                        2005
6 rows selected.

SQL> insert into owns values('6666','KA052005');

1 row created.

SQL> select * from owns;

DRIVERID                                  REGNO
---------------                             ---------
1111                                     KA04Q2301
2222                                     KA05P1000
3333                                     KA03L1234
4444                                     KA03L9999
5555                                     KA01P4026
6666                                     KA052005
6 rows selected.

SQL> insert into accident values (420,'30-Jan-2005','MGroad');

1 row created.

SQL> select * from accident;

 REPORTNO          ACCDATE                  LOCATION
 ----------          ---------               -------------
    12            01-JUN-01                Jayanagar
    25             02-JUL-02              Avenue Road
   512            08-MAR-00                 MG Road
  1024            25-OCT-02                Brigade Road
  1000            23-DEC-03               Richmond Circle
   420            30-JAN-05                   M G road
6 rows selected.
```

```
SQL> insert into participated values ('6666','KA052005', 420, 25000);

1 row created.

SQL> select * from participated;

DRIVERID            REGNO      REPORTNO                 DAMAGEAMOUNT
---------------        ---------      ----------                         ------------
1111         KA04Q2301          12                       25000
2222          KA05P1000         25                       15000
3333          KA03L1234         512                      15500
4444         KA03L9999         1024                       20000
5555         KA01P4026         1000                        5000
6666          KA052005          420                      25000
6 rows selected.
```

******* 3RD QUERY *******

```
SQL> select count(*) from Accident where Accdate like '__-___-02';

  COUNT(*)
----------
         2
```

******* 4TH QUERY *******

```
SQL> select count(*) from Car C,Participated P
          where C.Regno=P.Regno and C.Model='Ford';

 COUNT (*)
       ----------
          1
```

Order Processing Database

Consider the following relations for an order processing database applications in a Company

CUSTOMER (cust:int,cname:string,city:string)

ORDER (order:int,odate:date,cust:int,ord-amt:int)

ORDER-ITEM (order:int,item:int,qty:int)

ITEM (item:int,unitprice:int)

SHIPMENT (order:int,warehouse:int,ship-date:date)

WAREHOUSE (warehouse:int,city:string)

1) create the above tables by properly specifying the primary keys and the foreign keys
2) enter atleast five tuples for each relation
3) produce a listing: CUSTNAME,# of orders,AVG_ORDER_AMT,where the middle column is the total no of orders by the customer and the last column is the average order amount for that customer
4) list the order # for orders that were shipped from all warehouses that the company has in a specified city
5) Demonstrate how you delete item #10 from ITEM table and make the field null in the ORDER_ITEM table.

```
SQL> create table customer(
   Custno integer,
   Cname varchar(15) not null,
   City varchar(15),
   primary key (Custno));
Table created.
```

```
SQL> create table corder(
   Orderno integer,
   Odate date not null,
   Custno integer,
   OrderAmount integer not null,
   primary key (Orderno),
   foreign key (Custno) references Customer(Custno));
Table created.
```

```
SQL> create table item(
    itemno integer,
    unitprice integer not null,
    primary key (itemno));
Table created.

SQL> create table order_item(
orderno integer,
itemno integer,
quantity integer,
primary key (orderno,itemno),
foreign key (orderno) references corder(orderno),
foreign key (itemno) references item(itemno));
Table created.

SQL> create table warehouse(
  warehouseno integer,
  city varchar(15) not null,
  primary key (warehouseno));
Table created.

SQL> create table shipment(
   orderno integer,
   warehouseno integer,
   shipdate date,
   primary key (orderno,warehouseno),
   foreign key (orderno) references corder(orderno),
   foreign key (warehouseno) references warehouse(warehouseno));
Table created.

SQL> desc customer;
  SQL> insert into customer values(1111,'Jack','Bangalore');
1 row created.

SQL> insert into customer values(2222,'Fred','New York');

1 row created.

SQL> insert into customer values(3333,'George','Amsterdam');

1 row created.

SQL> insert into customer values(4444,'Kumar','Bangalore');

1 row created.

SQL> insert into customer values(5555,'Das','Bangalore');

1 row created.
```

```
SQL> select * from customer;

CUSTNO                    CNAME                          CITY
 ----------             ---------------            ---------------
 1111                    Jack                          Bangalore
 2222                      Fred                            New York
3333                    George                       Amsterdam
4444                    Kumar                         Bangalore
5555                      Das                          Bangalore
SQL> desc corder;

 Name                                  Null?                    Type
 --------------------------          --------         ---------------------------
 ORDERNO                            NOT NULL              NUMBER(38)
 ODATE                             NOT NULL                 DATE
 CUSTNO                                                    NUMBER(38)
 ORDERAMOUNT                        NOT NULL            NUMBER(38)
SQL> insert into corder values(1,'10-Mar-2004',1111,10000);

1 row created.

SQL> insert into corder values(2,'15-Apr-2005',2222,25000);

1 row created.

SQL> insert into corder values(3,'15-Dec-2004',3333,30000);

1 row created.

SQL> insert into corder values(4,'17-Jun-2004',4444,40000);

1 row created.

SQL> insert into corder values(5,'11-Jul-2004',2222,50000);

1 row created.

SQL> select * from corder;

ORDERNO        ODATE           CUSTNO               ORDERAMOUNT
----------        ---------         ----------               -----------
  1         10-MAR-04            1111                      10000
 2          15-APR-05           2222                      25000
 3          15-DEC-04             3333                      30000
4           17-JUN-04            4444                      40000
5           11-JUL-04            2222                      50000
```

```
SQL> desc item;
 Name                                  Null?                       Type
 -----------------------             --------          ----------------------------
 ITEMNO                               NOT NULL                 NUMBER(38)
 UNITPRICE                            NOT NULL                 NUMBER(38)
SQL> insert into item values(11,500);

1 row created.

SQL> insert into item values(22,250);

1 row created.

SQL> insert into item values(33,100);

1 row created.

SQL> insert into item values(44,50);

1 row created.

SQL> insert into item values(55,2);

1 row created.

SQL> insert into item values(10,150);

1 row created.

SQL> select * from item;

 ITEMNO                                        UNITPRICE
----------                                    ----------
   11                                              500
   22                                              250
   33                                              100
   44                                               50
   55                                                2
    10                                             150
6 rows selected.

SQL> desc order_item;

 Name                                          Null?                    Type
 ----------------------------              --------          ------------------------
 ORDERNO                                    NOT NULL                 NUMBER(38)
 ITEMNO                                     NOT NULL                 NUMBER(38)
 QUANTITY                                                            NUMBER(38)
SQL> insert into order_item values (1,11,150);

1 row created.
```

```
SQL> insert into order_item values (2,22,200);
1 row created.
SQL> insert into order_item values (3,33,300);
1 row created.
SQL> insert into order_item values (4,44,400);
1 row created.
SQL> insert into order_item values (5,55,500);
1 row created.
SQL> insert into order_item values(2,10,500);
1 row created.
SQL> select * from order_item;
ORDERNO                 ITEMNO                QUANTITY
----------             ----------              ----------
  1                       11                     150
  2                       22                      200
  3                       33                      300
  4                       44                      400
  5                       55                      500
  2                       10                      500
6 rows selected.
SQL> desc warehouse;
 Name                           Null?                  Type
 -------------------------     --------          --------------------------
 WAREHOUSENO                   NOT NULL                NUMBER(38)
 CITY                         NOT NULL            VARCHAR2(15)
SQL> insert into warehouse values(17,'Bangalore');
1 row created.
SQL> insert into warehouse values(27,'Chennai');
1 row created.
SQL> insert into warehouse values(37,'Pune');
1 row created.
SQL> insert into warehouse values(47,'Coimbatore');
1 row created.
```

```
SQL> insert into warehouse values(57,'Cochin');

1 row created.

SQL> select * from warehouse;

WAREHOUSENO                         CITY
-----------                      ---------------
  17                               Bangalore
  27                                Chennai
  37                                 Pune
  47                                 Coimbatore
  57                                  Cochin
SQL> desc shipment;

 Name                          Null?                Type
 -----------------------------  --------   ----------------------------
 ORDERNO                         NOT NULL            NUMBER(38)
 WAREHOUSENO                      NOT NULL             NUMBER(38)
 SHIPDATE                                                      DATE
SQL> insert into shipment values(1,17,'02-Jul-2005');

1 row created.

SQL> insert into shipment values(2,17,'15-Apr-2005');

1 row created.

SQL> insert into shipment values(3,27,'6-Jun-2005');

1 row created.

SQL> insert into shipment values(4,37,'10-May-2005');

1 row created.

SQL> insert into shipment values(5,47,'9-Feb-2005');

1 row created.

SQL> select * from shipment;

ORDERNO          WAREHOUSENO                      SHIPDATE
----------           -----------                          ---------
   1                    17                           02-JUL-05
   2                    17                          15-APR-05
   3                    27                           06-JUN-05
   4                    37                          10-MAY-05
   5                   47                            09-FEB-05
```

******* 1ST QUERY *******

```
SQL> select C.Cname,count(CO.orderno),Avg(CO.Orderamount)
    from Customer C,corder CO
    where C.custno=CO.custno
    group by C.Cname,CO.custno;
CNAME       COUNT(CO.ORDERNO)          AVG(CO.ORDERAMOUNT)
---------------      -----------------                -------------------
Fred                        2                                   37500
George                      1                                    30000
Jack                        1                                    10000
Kumar                       1                                   40000
```

******* 2ND QUERY *******

```
SQL> select orderno,warehouseno
    from shipment
    where warehouseno in
    (select warehouseno
     from warehouse
     where city='Bangalore');
  ORDERNO                      WAREHOUSENO
     ----------                        -----------
         1                              17
         2                              17
```

STUDENT DATABASE

Consider the following database of student enrollment in courses and books adopted for each course

STUDENT (regno:string,name:string,major:string,bdate:date)

COURSE (course:int,cname:string,dept:string)

ENROLL (regno:string,course:int,marks:int)

BOOK_ADOPTION (course:int,sem:int,book-ISBN:int)

TEXT (book-ISBN: int,book-title:string,publisher:string,author:string)

1) create the above tables by properly specifying the primary keys and foreign keys
2) enter five tuples for each relation
3) demonstrate how you add a new text book to the database and make this book be adopted by some department
4) produce a list of text books in alphabetical order for courses offered by CS department that use more than two books
5) list any department that has all its adopted books published by a specific publisher

```
SQL> create table Student(
   Regno varchar(10),
   Name varchar(10) not null,
   Major varchar(10) not null,
   Bdate date,
   primary key (Regno));
Table created.
```

```
SQL> create table Course(
   Courseno integer,
   Cname varchar(10) not null,
   Dept varchar(10) not null,
   primary key (Courseno));
Table created.
```

```
SQL> create table Enroll(
    Regno varchar(10),
    Courseno integer,
    Sem integer not null,
    Marks integer,
    primary key (Regno,Courseno),
    foreign key (Regno) references Student(Regno),
    foreign key (Courseno) references Course(Courseno));
Table created.
SQL> create table Text(
    ISBN integer,
    Booktitle varchar(20) not null,
    Publisher varchar(20),
    Author varchar(15),
    primary key (ISBN));
Table created.
SQL> create table Book_adoption(
    Courseno integer,
    Sem integer,
    ISBN integer,
    primary key (Courseno,Sem),
    foreign key (Courseno) references Course(Courseno),
    foreign key (ISBN) references Text(ISBN));
Table created.
SQL> desc student;
 Name                              Null?                     Type
 -------------------------------  --------     -------------------------
 REGNO                            NOT NULL           VARCHAR2(10)
 NAME                              NOT NULL           VARCHAR2(10)
 MAJOR                              NOT NULL          VARCHAR2(10)
 BDATE                                                          DATE
```

```
SQL> insert into Student values('1BI02CS010','Karan','CSE','02-Jan-1984');

1 row created.

SQL> insert into Student values('1BI02EE015','Jack','EEE','15-Apr-1983');

1 row created.

SQL> insert into Student values('1BI00CS010','Adi','CSE','02-Jan-1982');

1 row created.

SQL> insert into Student values('1BI01EC089','Rahul','ECE','01-Dec-1983');

1 row created.

SQL> insert into student values ('1BI01ME075','Sachin','MECH','18-Jul-1983');

1 row created.

SQL> select * from student;

REGNO         NAME          MAJOR         BDATE
----------    ----------    ----------    ---------
1BI01ME075    Sachin        MECH          18-JUL-83
1BI02CS010    Karan         CSE           02-JAN-84
1BI02EE015    Jack          EEE           15-APR-83
1BI00CS010    Adi           CSE           02-JAN-82
1BI01EC089    Rahul         ECE           01-DEC-83
SQL> desc course;

 Name                                      Null?       Type
 ----------------------------------------- --------    ----------------------
 COURSENO                                  NOT NULL    NUMBER(38)
 CNAME                                     NOT NULL    VARCHAR2(10)
 DEPT                                      NOT NULL    VARCHAR2(10)
SQL> insert into course values(11,'DSC','CSE');

1 row created.

SQL> insert into course values(22,'ADA','CSE');

1 row created.

SQL> insert into course values(33,'CN','EC');

1 row created.

SQL> insert into course values(44,'TD','MECH');

1 row created.
```

```
SQL> insert into course values(55,'MP','EC');

1 row created.

SQL> select * from course;

 COURSENO            CNAME            DEPT
  ----------         ----------       ----------
     11               DSC              CSE
     22               ADA              CSE
     33               CN               EC
     44               TD               MECH
     55               MP               EC
SQL> desc enroll;

 Name                           Null?               Type
 ------------------------      --------      --------------------------
 REGNO                         NOT NULL          VARCHAR2(10)
 COURSENO                     NOT NULL          NUMBER(38)
 SEM                          NOT NULL         NUMBER(38)
 MARKS                                        NUMBER(38)
SQL> insert into enroll values('1BI02CS010',22,5,72);

1 row created.

SQL> insert into enroll values('1BI00CS010',11,3,90);

1 row created.

SQL>  insert into enroll values('1BI01EC089',33,6,52);

1 row created.

SQL> insert into enroll values('1BI01ME075',44,4,85);

1 row created.

SQL> insert into enroll values('1BI02EE015',22,5,75);

1 row created.

SQL> select * from enroll;

REGNO         COURSENO               SEM            MARKS
----------     ----------            ----------       ----------
1BI02CS010          22                 5             72
1BI00CS010          11                 3             90
1BI01EC089          33                 6             52
1BI01ME075         44                  4             85
1BI02EE015          22                  5            75
```

```
SQL> desc text;
 Name                                  Null?              Type
 ---------------------------------    --------        --------------------------
 ISBN                               NOT NULL             NUMBER(38)
 BOOKTITLE                          NOT NULL             VARCHAR2(20)
 PUBLISHER                                             VARCHAR2(20)
 AUTHOR                                                VARCHAR2(15)
SQL> insert into text values(7722,'VB6','Dreamtech','Holzner');

1 row created.

SQL> insert into text values(1144,'DS with C','Sapna','Nandagopalan');

1 row created.

SQL> insert into text values(4400,'C programming', 'TMH', 'Balaguruswamy');

1 row created.

SQL> insert into text values(5566,'Computer Nw','PHI','Tennenbaum');

1 row created.

SQL> insert into text values(3388,'MP','PHI','Brey');

1 row created.

SQL> select * from text;
  ISBN      BOOKTITLE               PUBLISHER               AUTHOR
----------   --------------------    --------------------    ---------------
 7722       VB6                     Dreamtech                   Holzner
1144      DS with C                   Sapna                Nandagopalan
4400  C Programming                  TMH                 Balaguruswamy
5566   Computer Nw                   PHI                   Tennenbaum
3388        MP                      PHI                       Brey
SQL> desc book_adoption;
 Name                                      Null?         Type
 ---------------------                    --------     ------------------------
 COURSENO                                 NOT NULL        NUMBER(38)
 SEM                                      NOT NULL        NUMBER(38)
 ISBN                                                    NUMBER(38)
SQL> insert into book_adoption values(11,3,7722);

1 row created.

SQL> insert into book_adoption values(22,4,7722);

1 row created.
```

```
SQL> insert into book_adoption values(11,5,4400);
1 row created.
SQL> insert into book_adoption values(11,8,5566);
1 row created.
SQL> insert into book_adoption values(55,4,3388);
1 row created.
SQL> insert into book_adoption values(44,4,5566);
1 row created.
SQL> insert into book_adoption values(44,7,3388);
1 row created.
SQL> select * from book_adoption;
 COURSENO                          SEM           ISBN
  ----------                     ----------    ----------
     11                          3             7722
     22                          4             7722
     11                          5             4400
     11                          8             5566
     55                          4             3388
     44                          4             5566
     44                          7             3388
7 rows selected.
```

******* 1ST QUERY *******

```
SQL> insert into text values(1234,'Elec.Circuits','Sapna','Giridhar');
1 row created.
SQL> insert into book_adoption values(55,3,1234);
1 row created.
SQL> select * from text;
ISBN          BOOKTITLE           PUBLISHER         AUTHOR
----------     --------------------   --------------------    ---------------7722
VB6              Dreamtech            Holzner
1144            DS with C             Sapna             Nandagopalan
 4400         C Programming          TMH            Balaguruswamy
 5566           Computer Nw          PHI           Tennenbaum
 3388               MP               PHI                 Brey
 1234           Elec.Circuits         Sapna             Giridhar
6 rows selected.
```

```
SQL> select * from book_adoption;
  COURSENO             SEM         ISBN
     ----------        ----------  ----------
        11              3          7722
        22              4          7722
        11              5          4400
        11              8          5566
        55              4          3388
        44              4          5566
        44              7          3388
        55              3          1234
8 rows selected.
```

***** 2ND QUERY *****

```
SQL> select C.Courseno,T.ISBN,T.Booktitle
  from Course C,Book_adoption BA,Text T
  where C.Courseno=BA.Courseno and BA.ISBN=T.ISBN and C.Dept='CSE'
  group by C.Courseno,T.ISBN,T.Booktitle;
  COURSENO           ISBN        BOOKTITLE
    ----------        ----------  --------------------
        11            4400        C Programming
        11            5566        Computer Nw
        11            7722             VB6
        22            7722             VB6
```

***** 3RD QUERY *****

```
SQL>  select distinct C.Dept
  from Course C, Book_adoption A,Text T
  where C.Courseno=A.Courseno and
  A.ISBN=T.ISBN and
  not exists (( select Y.ISBN
  from Course X,Book_Adoption Y
  where X.Courseno=Y.Courseno
  and X.Dept=C.Dept)
  minus
 (select ISBN
 from Text
 where publisher='PHI'));
DEPT
----------
MECH
```

BOOK DEALER DATABASE

The following tables are maintained by a book dealer

AUTHOR(author-id:int,name:string,city:string,country:string)

PUBLISHER(publisher-id:int,name:string,city:string,country:string)

CATALOG(book-id:int,title:string,author-id:int,publisher-id:int,category-id:int,year:int,price:int)

CATEGORY(category-id:int,description:script)

ORDER-DETAILS(order-no:int,book-id:int,quantity:int)

1) create the above details by properly specifying the primary keys and foreign keys
2) enter atleast five tuples for each relation
3) find the author of the book which has maximium sales
4) demonstrate how you increase the price of books published by a specific publisher by 10%
5) generation of suitable reports
6) create suitable front end for querying and display the results

```
SQL> create table Author(
    Authorid integer,
    Aname varchar(15),
    Acity varchar(15),
    Acountry varchar(15),
    primary key (Authorid));
Table created.
SQL> create table Publisher(
    Publisherid integer,
    Pname varchar(15),
    Pcity varchar(15),
    Pcountry varchar(15),
    primary key (Publisherid));
Table created.
```

```
SQL> create table Category(
    Categoryid integer,
    Description varchar(20),
    primary key (Categoryid));
Table created.
SQL> create table Catalog(
    Bookid integer,
    Title varchar(20),
    Authorid integer,
    Publisherid integer,
    Categoryid integer,
    Year integer,
    Price integer,
    primary key (Bookid),
   foreign key (Authorid) references Author(Authorid),
   foreign key (Publisherid) references Publisher(Publisherid),
   foreign key (Categoryid) references Category(Categoryid));
Table created.
SQL> create table Order_details(
    Orderno integer,
    Bookid integer,
    Quantity integer,
    primary key (Orderno,Bookid),
    foreign key (Bookid) references Catalog(Bookid));
Table created.
SQL> desc author;
 Name                       Null?                     Type
 ---------------------     ----- ---          ----------------------------
 AUTHORID                 NOT NULL                NUMBER(38)
 ANAME                                           VARCHAR2(15)
 ACITY                                            VARCHAR2(15)
 ACOUNTRY                                        VARCHAR2(15)
```

```
SQL> insert into Author values(1000,'Nandagopalan','Bangalore','India');

1 row created.

SQL> insert into Author values(2000,'Tony','Haywood','USA');

1 row created.

SQL> insert into Author values(3000,'Holzner','New York','USA');

1 row created.

SQL> insert into Author values(4000,'Tennenbaum','London','UK');

1 row created.

SQL> insert into Author values(5000,'Balaguruswamy','Chennai','India');

1 row created.

SQL> select * from Author;

 AUTHORID         ANAME           ACITY          ACOUNTRY
 ----------    ---------------   ---------------   ---------------
   1000         Nandagopalan       Bangalore              India
  2000              Tony            Haywood                USA
 3000             Holzner          New York               USA
 4000            Tennenbaum          London                 UK
 5000          Balaguruswamy         Chennai               India
SQL> desc publisher;

 Name                                Null?                 Type
 ------------------------------    --------         ---------------------
 PUBLISHERID                      NOT NULL              NUMBER(38)
 PNAME                                                 VARCHAR2(15)
 PCITY                                                 VARCHAR2(15)
 PCOUNTRY                                              VARCHAR2(15)
SQL> insert into publisher values(11,'Wiely','NewDelhi','India');

1 row created.

SQL> insert into publisher values(22,'PHI','California','USA');

1 row created.

SQL> insert into publisher values(33,'Sapna','Bangalore','India');

1 row created.

SQL> insert into publisher values(44,'TMH','NewYork','USA');

1 row created.
```

```
SQL> insert into publisher values(55,'Wrox','Texas','USA');

1 row created.

SQL> select * from publisher;

PUBLISHERID        PNAME           PCITY          PCOUNTRY
-----------       ---------------   ---------------   ---------------
 11                    Wiely         NewDelhi                India
 22                     PHI           California              USA
 33                   Sapna          Bangalore               India
 44                   TMH          NewYork                 USA
 55                    Wrox             Texas                 USA
SQL> desc category;

 Name                              Null?                   Type
 ----------------------------      --------          ------------------------
 CATEGORYID                       NOT NULL              NUMBER(38)
 DESCRIPTION                                            VARCHAR2(20)
SQL> insert into category values(1,'OS');

1 row created.

SQL> insert into category values(2,'Languages');

1 row created.

SQL> insert into category values(3,'Hardware');

1 row created.

SQL> insert into category values(4,'Algorithms');

1 row created.

SQL> insert into category values(5,'Internet');

1 row created.

SQL> select * from category;

  CATEGORYID                    DESCRIPTION
   ----------                   --------------------
       1                            OS
       2                          Languages
       3                           Hardware
       4                          Algorithms
       5                          Internet
```

```
SQL> desc catalog;
 Name                                     Null?          Type
 -----------------------------------    --------      ----------------------------
 BOOKID                                   NOT NULL          NUMBER(38)
 TITLE                                                      VARCHAR2(20)
 AUTHORID                                                     NUMBER(38)
 PUBLISHERID                                                  NUMBER(38)
 CATEGORYID                                                   NUMBER(38)
 YEAR                                                         NUMBER(38)
 PRICE                                                        NUMBER(38)
SQL> insert into catalog values(123,'DSC',1000,33,2,2000,185);

1 row created.

SQL> insert into catalog values(456,'Networks',4000,44,4,2002,365);

1 row created.

SQL> insert into catalog values(789,'VB6',2000,11,2,2000,300);

1 row created.

SQL> insert into catalog values (213,'Frontpage2002',4000,44,5,2003,500);

1 row created.

SQL> insert into catalog values(879,'ADA',1000,33,4,2001,195);

1 row created.

SQL> select * from catalog;

BOOKID TITLE AUTHORID PUBLISHERID CATEGORYID   YEAR   PRICE ----
----    -------   ----------     -----------      ----------      --------  ---------
123       DSC           1000       33                  2        2000        185
 456     Networks      4000       44                  4        2002        365
789         VB6           2000     11                  2        2000        300
213   Frontpage2002  4000      44                  5        2003        500
879       ADA          1000     33                  4        2001        195
SQL> desc order_details;
 Name                                Null?                  Type
 --------------------------         --------          -----------------------
 ORDERNO                      NOT NULL                NUMBER(38)
 BOOKID                        NOT NULL                NUMBER(38)
 QUANTITY                                              NUMBER(38)
SQL> insert into order_details values(112,123,100);

1 row created.
```

```
SQL> insert into order_details values(113,123,20);
1 row created.
SQL> insert into order_details values(114,213,50);
1 row created.
SQL> insert into order_details values(115,789,500);
1 row created.
SQL> insert into order_details values(116,879,8);
1 row created.
SQL> select * from order_details;
  ORDERNO                    BOOKID                  QUANTITY
   ----------                ----------              ----------
     112                      123                      100
     113                      123                     20
     114                      213                     50
     115                      789                    500
     116                      879                     8
```

***** 1ST QUERY *****

```
SQL> select C.Authorid,A.Aname
 from Catalog C,Author A
 where   A.Authorid=C.Authorid   and   C.Year>2000   and   C.Price>(Select
Avg(Price) from Catalog)
 group by C.Authorid,A.Aname
 having count(C.Authorid)>=2;
 AUTHORID          ANAME
  ----------        ---------------
   4000             Tennenbaum
```

***** 2ND QUERY *****

```
SQL> create view salesdetails as(
   Select OD.Bookid as Book#,C.Price as Cost,Sum(OD.quantity) as Qty,
   sum(OD.quantity*C.price) as sales
   from Order_details OD,Catalog C,Author A
   where OD.Bookid=C.Bookid and C.Authorid=A.Authorid
   group by OD.Bookid,C.Price);
View created.
```

```
SQL> select A.Authorid,A.Aname,S.Book#,S.Sales
    from Author A,Catalog C,Salesdetails S
    where A.Authorid=C.Authorid and S.Book#=C.Bookid and sales=(
    select Max(Sales) from Salesdetails);
AUTHORID            ANAME            BOOK#      SALES
   ----------          ---------------       ----------    ----------
      2000               Tony             789        150000
```

******* 3RD QUERY *******

BEFORE:

```
SQL> select * from Catalog;
BOOKID TITLE AUTHORID PUBLISHERID CATEGORYID   YEAR   PRICE
--------   ----------   ----------     -----------        ----------      -------- ---------
 123        DSC        1000        33                2          2000         185
 456     Networks     4000     44                    4          2002         365
789          VB6    2000         11                  2          2000         300
 213   Frontpage2002     4000       44              5            2003        500
879        ADA       1000        33              4            2001         195
SQL> update catalog
   set price=price*1.10
  where publisherid=33;
```

2 rows updated.

AFTER:

```
SQL> select * from catalog;
BOOKID TITLE AUTHORID PUBLISHERID CATEGORYID   YEAR   PRICE
------     ----------          -----------           ----------       ---------- -----------
123      DSC        1000            33                2          2000         204
456     Networks       4000        44                4          2002         365
789       VB6               2000          11              2          2000         300 213
Frontpage2002      4000      44                  5              2003         500  879
ADA          1000              33         4            2001         215
```

BANKING DATABASE

Consider the following database for a banking enterprise

BRANCH(branch-name:string,branch-city:string,assets:real)

ACCOUNT(accno:int,branch-name:string,balance:real)

DEPOSITOR(customer-name:string,accno:int)

CUSTOMER(customer-name:string,customer-street:string,city:string)

LOAN(loan-number:int,branch-name:string,loan-number-int)

BORROWER(customer-name:string,customer-street:string,city:string)

1) create the above tables by properly specifying the primary and foreign keys
2) enter 5 tuples for each relation
3) find all the customers who have atleast two accounts at the main branch
4) find all the customers who have an account at all the branches located in a specified city
5) demonstrate how you delete all account tuples at every branch located in a specified city
6) genetration of suitable reports
7) create suitable front end for querying and display the results

```
SQL> create table branch(
  branchname varchar(15),
  branchcity varchar(15),
  assets real,
  primary key (branchname));
Table created.
```

```
SQL> create table accnt(
   accountno integer,
  branchname varchar(15),
   balance real,
  primary key (accountno),
   foreign key (branchname) references branch(branchname));

Table created.
```

```
SQL> create table depositor(
  customername varchar(15),
  accountno integer,
  primary key (customername,accountno),
  foreign key (accountno) references accnt(accountno));
Table created.

SQL> create table custmer(
  customername varchar(15),
  customerstreet varchar(15),
  city varchar(15),
  primary key (customername));
Table created.

SQL> create table loan(
  loanno integer,
  branchname varchar(15),
  amount real,
  primary key (loanno),
  foreign key (branchname) references branch(branchname));
Table created.

SQL> create table borrower(
  customername varchar(15),
  loanno integer,
  primary key (customername,loanno),
  foreign key (customername) references custmer(customername),
  foreign key (loanno) references loan(loanno));
Table created.

SQL> desc branch;

 Name                                          Null?              Type

 --------------------------                  --------        ----------------------------

 BRANCHNAME                                    NOT NULL         VARCHAR2(15)

 BRANCHCITY                                                     VARCHAR2(15)

 ASSETS                                                          NUMBER(63)

SQL> insert into branch values('Jayanagar','Bangalore','15000000');

1 row created.

SQL> insert into branch values('Basavanagudi','Bangalore','25000000');

1 row created.
```

```
SQL> insert into branch values('Noida','NewDelhi','50000000');

1 row created.

SQL> insert into branch values('Marinedrive','Mumbai','40000000');

1 row created.

SQL> insert into branch values('GreenPark','Newdelhi','30000000');

1 row created.

SQL> select * from branch;

BRANCHNAME                     BRANCHCITY            ASSETS
 ---------------               ---------------       ----------
Jayanagar                        Bangalore             15000000
Basavanagudi                    Bangalore            25000000
Noida                          NewDelhi            50000000
Marinedrive                     Mumbai               40000000
GreenPark                      Newdelhi             30000000
SQL> desc accnt;

 Name                          Null?                      Type
 -------------------------    --------                   ---------------
 ACCOUNTNO                 NOT NULL                       NUMBER(38)
 BRANCHNAME                                            VARCHAR2(15)
 BALANCE                                                  NUMBER(63)
SQL> insert into accnt values('123','Jayanagar','25000');

1 row created.

SQL> insert into accnt values('156','Jayanagar','30000');

1 row created.

SQL> insert into accnt values('456','Basavanagudi','15000');

1 row created.

SQL> insert into accnt values('789','Noida','25000');

1 row created.

SQL> insert into accnt values('478','Marinedrive','48000');

1 row created.

SQL> insert into accnt values('778','GreenPark','60000');

1 row created.

SQL> insert into accnt values('189','Basavanagudi','48888');

1 row created.
```

```
SQL> select * from accnt;
 ACCOUNTNO                        BRANCHNAME                   BALANCE
----------                        ---------------            ----------
     123                            Jayanagar                   25000
     156                            Jayanagar                   30000
     456                          Basavanagudi                 15000
     789                              Noida                    25000
     478                           Marinedrive                 48000
     778                            GreenPark                  60000
     189                           Basavanagudi                48888
7 rows selected.
SQL> desc custmer;
 Name                                  Null?                        Type
 ---------------------------           --------          ----------------------
 CUSTOMERNAME                          NOT NULL               VARCHAR2(15)
 CUSTOMERSTREET                                               VARCHAR2(15)
 CITY                                                        VARCHAR2(15)
SQL> insert into custmer values('Ramu','Jayanagar','Bangalore');
1 row created.
SQL> insert into custmer values('Kumar','Basavanagudi','Bangalore');
1 row created.
SQL> insert into custmer values('John','Noida','Newdelhi');
1 row created.
SQL> insert into custmer values('Mike','Marinedrive','Mumbai');
1 row created.
SQL> insert into custmer values('Sachin','GreenPark','NewDelhi');
1 row created.
SQL> select * from custmer;
CUSTOMERNAME                     CUSTOMERSTREET               CITY
---------------                    ---------------            ---------------
Ramu                                Jayanagar                   Bangalore
Kumar                             Basavanagudi                  Bangalore
John                                  Noida                      Newdelhi
Mike                               Marinedrive                    Mumbai
Sachin                              GreenPark                    NewDelhi
```

```
SQL> desc depositor;
 Name                                      Null?                Type
 ----------------------------------        --------     ----------------------------
 CUSTOMERNAME                               NOT NULL        VARCHAR2(15)
 ACCOUNTNO                                   NOT NULL           NUMBER(38)
SQL> insert into depositor values('Ramu',123);
1 row created.
SQL> insert into depositor values('Ramu',156);

1 row created.

SQL> insert into depositor values('Ramu',189);

1 row created.

SQL> insert into depositor values('Kumar',456);

1 row created.

SQL> insert into depositor values('John',789);

1 row created.

SQL>  insert into depositor values('Mike',478);

1 row created.

SQL>  insert into depositor values('Sachin',778);

1 row created.

SQL> select * from depositor;

CUSTOMERNAME                        ACCOUNTNO
---------------                        ----------
Ramu                                     123
Ramu                                     156
Ramu                                     189
Kumar                                   456
John                                     789
Mike                                    478
Sachin                                  778
7 rows selected.

SQL> desc loan;

 Name                                      Null?            Type
 ------------------------------------        --------     ---------------------------
 LOANNO                                    NOT NULL           NUMBER(38)
 BRANCHNAME                                                VARCHAR2(15)
 AMOUNT                                                          NUMBER(63)
```

```
SQL> insert into loan values('1111','Jayanagar','250000');

1 row created.

SQL> insert into loan values('2222','Basavanagudi','350000');

1 row created.

SQL>  insert into loan values('3333','Noida','150000');

1 row created.

SQL> insert into loan values('4444','Marinedrive','1500000');

1 row created.

SQL> insert into loan values('5555','GreenPark','7500000');

1 row created.

SQL> select * from loan;

   LOANNO                 BRANCHNAME              AMOUNT
  ----------             ---------------         ----------
     1111                   Jayanagar               250000
     2222                 Basavanagudi             350000
     3333                     Noida                150000
     4444                  Marinedrive            1500000
     5555                   GreenPark              7500000
SQL> desc borrower;

 Name                                Null?          Type
 --------------------------------   --------     ---------------------------
 CUSTOMERNAME                       NOT NULL        VARCHAR2(15)
 LOANNO                             NOT NULL           NUMBER(38)
SQL> insert into borrower values('Ramu',1111);

1 row created.

SQL> insert into borrower values('Kumar',2222);

1 row created.

SQL> insert into borrower values('John',3333);

1 row created.

SQL> insert into borrower values('Mike',4444);

1 row created.

SQL> insert into borrower values('Sachin',5555);

1 row created.
```

```
SQL> select * from borrower;

CUSTOMERNAME                 LOANNO
---------------                ----------
Ramu                              1111
Kumar                            2222
John                              3333
Mike                              4444
Sachin                            5555
```

******* 1ST QUERY *******

```
SQL> select customername
   from Depositor D,Accnt A
  where D.accountno=A.accountno and
  branchname='Jayanagar'
  having count(D.accountno)>=2
  group by customername;
CUSTOMERNAME
---------------
Ramu
```

******* 2ND QUERY *******

```
SQL> select customername
  from Branch B,Accnt A,Depositor D
  where B.Branchname=A.Branchname and
   A.Accountno=D.Accountno and
  B.Branchcity='Bangalore'
  having count(distinct B.Branchname)=(Select
   count(Branchname)
 from Branch
  where Branchcity='Bangalore')
 group by customername;

CUSTOMERNAME
---------------
Ramu
```

******* 3RD QUERY *******

```
SQL>delete from ACCOUNT
where BNAME in ( select        BNAME
                    from  BRANCH
                    where        BCITY = '&CITY' ) ;
```