Database solutions

Sociology, Communication

DATABASE SOLUTIONS (2nd Edition) THOMAS M CONNOLLY & CAROLYN E BEGG SOLUTIONS TO REVIEW QUESTIONS Chapter 1 Introduction- Review questions 1. 1List four examples of database systems other than those listed in Section 1. 1. Some examples could be: •A system that maintains component part details for a car manufacturer; •An advertising company keeping details of all clients and adverts placed with them; •A training company keeping course information and participants' details; •An organization maintaining all sales order information. 1. 2Discuss the meaning of each of the following terms: (a)data

For end users, this constitutes all the different values connected with the various objects/entities that are of concern to them. (b)database A shared collection of logically related data (and a description of this data), designed to meet the information needs of an organization. (c)database management system A software system that: enables users to define, create, and maintain the database and provides controlled access to this database. (d)application program A computer program that interacts with the database by issuing an appropriate request (typically an SQL statement) to the DBMS. (e)data independence

This is essentially the separation of underlying file structures from the programs that operate on them, also called program-data independence. (f)views. A virtual table that does not necessarily exist in the database but is generated by the DBMS from the underlying base tables whenever it's accessed. These present only a subset of the database that is of particular interest to a user. Views can be customized, for example, field names may change, and they also provide a level of security preventing users from seeing certain data. 1. 3Describe the main characteristics of the database approach.

Focus is now on the data first, and then the applications. The structure of the data is now kept separate from the programs that operate on the data. This is held in the system catalog or data dictionary. Programs can now share data, which is no longer fragmented. There is also a reduction in redundancy, and achievement of program-data independence. 1. 4Describe the five components of the DBMSenvironmentand discuss how they relate to each other. (1)Hardware: The computer system(s) that the DBMS and the application programs run on. This can range from a single PC, to a single mainframe, to a network of computers. 2)Software: The DBMS software and the application programs, together with the operating system, including network software if the DBMS is being used over a network. (3)Data: The data acts as a bridge between the hardware and software components and the human components. As we’ve already said, the database contains both the operational data and the meta-data (the ‘ data about data’). (4)Procedures: The instructions and rules that govern the design and use of the database. This may include instructions on how to log on to the DBMS, make backup copies of the database, and how to handle hardware or software failures. 5)People: This includes the database designers, database administrators (DBAs), application programmers, and the end-users. 1. 5Describe the problems with the traditional two-tier client-server architecture and discuss how these problems were overcome with the three-tier client-server architecture. In the mid-1990s, as applications became more complex and potentially could be deployed to hundreds or thousands of end-users, the client side of this architecture gave rise to two problems: •A ‘ fat’ client, requiring considerable resources on the client’s computer to run effectively (resources include disk space, RAM, and CPU power). A significant client side administration overhead. By 1995, a new variation of the traditional two-tier client-server model appeared to solve these problems called the three-tier client-server architecture. This new architecture proposed three layers, each potentially running on a different platform: (1)The user interface layer, which runs on the end-user’s computer (the client). (2)The business logic and data processing layer. This middle tier runs on a server and is often called the application server. One application server is designed to serve multiple clients. (3)A DBMS, which stores the data required by the middle tier.

This tier may run on a separate server called the database server. The three-tier design has many advantages over the traditional two-tier design, such as: •A ‘ thin’ client, which requires less expensive hardware. •Simplified application maintenance, as a result of centralizing the business logic for many end-users into a single application server. This eliminates the concerns of software distribution that are problematic in the traditional two-tier client-server architecture. •Added modularity, which makes it easier to modify or replace one tier without affecting the other tiers. Easier load balancing, again as a result of separating the core business logic from the database functions. For example, a Transaction Processing Monitor (TPM) can be used to reduce the number of connections to the database server. (A TPM is a program that controls data transfer between clients and servers in order to provide a consistent environment for Online Transaction Processing (OLTP).) An additional advantage is that the three-tier architecture maps quite naturally to the Web environment, with a Web browser acting as the ‘ thin’ client, and a Web server acting as the application server.

The three-tier client server architecture is illustrated in Figure 1. 4. 1. 6Describe the functions that should be provided by a modern full-scale multi-user DBMS. Data Storage, Retrieval and UpdateAuthorization Services A User-Accessible CatalogSupport for DataCommunicationTransaction SupportIntegrity Services Concurrency Control ServicesServices to Promote Data Independence Recovery ServicesUtility Services 1. 7Of the functions described in your answer to Question 1. 6, which ones do you think would not be needed in a standalone PC DBMS? Provide justification for your answer.

Concurrency Control Services - only single user. Authorization Services - only single user, but may be needed if different individuals are to use the DBMS at different times. Utility Services - limited in scope. Support for Data Communication - only standalone system. 1. 8Discuss the advantages and disadvantages of DBMSs. Some advantages of the database approach include control of data redundancy, data consistency, sharing of data, and improved security and integrity. Some disadvantages include complexity, cost, reduced performance, and higher impact of afailure.

Chapter 2 The Relational Model - Review questions 2. 1Discuss each of the following concepts in the context of the relational data model: (a)relation A table with columns and rows. (b)attribute A named column of a relation. (c)domain The set of allowable values for one or more attributes. (d)tuple A record of a relation. (e)relational database. A collection of normalized tables.

2. 2Discuss the properties of a relational table. A relational table has the following properties: •The table has a name that is distinct from all other tables in the database. •Each cell of the table contains exactly one value. For example, it would be wrong to store several telephone numbers for a single branch in a single cell. In other words, tables don't contain repeating groups of data. A relational table that satisfies this property is said to be normalized or in first normal form.) •Each column has a distinct name. •The values of a column are all from the same domain. •The order of columns has no significance. In other words, provided a column name is moved along with the column values, we can interchange columns. •Each record is distinct; there are no duplicate records. The order of records has no significance, theoretically. 2. 3Discuss the differences between the candidate keys and the primary key of a table. Explain what is meant by a foreign key. How do foreign keys of tables relate to candidate keys? Give examples to illustrate your answer. The primary key is the candidate key that is selected to identify tuples uniquely within a relation. A foreign key is an attribute or set of attributes within one relation that matches the candidate key of some (possibly the same) relation. 2. 4What does a null represent?

Represents a value for a column that is currently unknown or is not applicable for this record. 2. 5Define the two principal integrity rules for the relational model. Discuss why it is desirable to enforce these rules. Entity integrityIn a base table, no column of a primary key can be null. Referential integrityIf a foreign key exists in a table, either the foreign key value must match a candidate key value of some record in its home table or the foreign key value must be wholly null. Chapter 3 SQL and QBE - Review questions 3.

1What are the two major components of SQL and what function do they serve?

A data definition language (DDL) for defining the database structure. A data manipulation language (DML) for retrieving and updating data. 3. 2Explain the function of each of the clauses in the SELECT statement. What restrictions are imposed on these clauses? FROMspecifies the table or tables to be used; WHEREfilters the rows subject to some condition; GROUP BYforms groups of rows with the same column value; HAVINGfilters the groups subject to some condition; SELECTspecifies which columns are to appear in the output; ORDER BYspecifies the order of the output. 3. What restrictions apply to the use of the aggregate functions within the SELECT statement? How do nulls affect the aggregate functions? An aggregate function can be used only in the SELECT list and in the HAVING clause. Apart from COUNT(*), each function eliminates nulls first and operates only on the remaining non-null values. COUNT(*) counts all the rows of a table, regardless of whether nulls or duplicate values occur. 3. 4Explain how the GROUP BY clause works. What is the difference between the WHERE and HAVING clauses? SQL first applies the WHERE clause. Then it conceptually arranges he table based on the grouping column(s). Next, applies the HAVING clause and finally orders the result according to the ORDER BY clause. WHERE filters rows subject to some condition; HAVING filters groups subject to some condition. 3. 5What is the difference between a subquery and a join? Under what circumstances would you not be able to use a subquery? With a subquery, the columns specified in the SELECT list are restricted to one table. Thus, cannot use a subquery if the SELECT list contains columns from more than one table. 3. 6What is QBE and what is the relationship between QBE and SQL?

QBE is an alternative, graphical-based, ‘ point-and-click’ way of querying the database, which is particularly suited for queries that are not too complex, and can be expressed in terms of a few tables. QBE has acquired the reputation of being one of the easiest ways for non-technical users to obtain information from the database. QBE queries are converted into their equivalent SQL statements before transmission to the DBMS server. Chapter 4 Database Systems Development Lifecycle - Review questions 4. 1Describe what is meant by the term ‘ software crisis’.

The past few decades has witnessed the dramatic rise in the number of software applications. Many of these applications proved to be demanding, requiring constant maintenance. This maintenance involved correcting faults, implementing new user requirements, and modifying the software to run on new or upgraded platforms. With so much software around to support, the effort spent on maintenance began to absorb resources at an alarming rate. As a result, many major software projects were late, over budget, and the software produced was unreliable, difficult to maintain, and performed poorly.

This led to what has become known as the ‘ software crisis’. Although this term was first used in the late 1960s, more than 30 years later, the crisis is still with us. As a result, some people now refer to the software crisis as the ‘ softwaredepression’. 4. 2Discuss the relationship between the information systems lifecycle and the database system development lifecycle. An information system is the resources that enable the collection, management, control, and dissemination of data/information throughout a company. The database is a fundamental component of an information system.

The lifecycle of an information system is inherently linked to the lifecycle of the database that supports it. Typically, the stages of the information systems lifecycle include: planning, requirements collection and analysis, design (including database design), prototyping, implementation, testing, conversion, and operational maintenance. As a database is a fundamental component of the larger company-wide information system, the database system development lifecycle is inherently linked with the information systems lifecycle. 4. 3Briefly describe the stages of the database system development lifecycle. See Figure 4. Stages of the database system development lifecycle. Database planning is the management activities that allow the stages of the database system development lifecycle to be realized as efficiently and effectively as possible. System definition involves identifying the scope and boundaries of the database system including its major user views. A user view can represent a job role or business application area. Requirements collection and analysis is the process of collecting and analyzing information about the company that is to be supported by the database system, and using this information to identify the requirements for the new system.

There are three approaches to dealing with multiple user views, namely the centralized approach, the view integration approach, and a combination of both. The centralized approach involves collating the users’ requirements for different user views into a single list of requirements. A data model representing all the user views is created during the database design stage.

The view integration approach involves leaving the users' requirements for each user view as separate lists of requirements. Data models representing each user view are created and then merged at a later stage of database design.

Database design is the process of creating a design that will support the company's mission statement and mission objectives for the required database. This stage includes the logical and physical design of the database. The aim of DBMS selection is to select a system that meets the current and future requirements of the company, balanced against costs that include the purchase of the DBMS product and any additional software/hardware, and the costs associated with changeover and training. Application design involves designing the user interface and the application programs that use and process the database.

This stage involves two main activities: transaction design and user interface design. Prototyping involves building a working model of the database system, which allows the designers or users to visualize and evaluate the system. Implementation is the physical realization of the database and application designs. Data conversion and loading involves transferring any existing data into the new database and converting any existing applications to run on the new database. Testing is the process of running the database system with the intent of finding errors.

Operational maintenance is the process of monitoring and maintaining the system following installation. 4. 4Describe the purpose of creating a mission statement and mission objectives for the required database during the database planning stage. The mission statement defines the major aims of the database system, while each mission objective identifies a particular task that the database must support. 4. 5Discuss what a user view represents when designing a database system. A user view defines what is required of a database system from the perspective of a particular job (such as Manager or

Supervisor) or business application area (such as marketing, personnel, or stock control). 4. 6Compare and contrast the centralized approach and view integration approach to managing the design of a database system with multiple user views. An important activity of the requirements collection and analysis stage is deciding how to deal with the situation where there is more than one user view. There are three approaches to dealing with multiple user views: •the centralized approach, •the view integration approach, and •a combination of both approaches.

Centralized approach Requirements for each user view are merged into a single list of requirements for the new database system. A logical data model representing all user views is created during the database design stage. The centralized approach involves collating the requirements for different user views into a single list of requirements. A data model representing all user views is created in the database design stage. A diagram representing the management of user views 1 to 3 using the centralized approach is shown in Figure 4. 4.

Generally, this approach is preferred when there is a significant overlap in requirements for each user view and the database system is not overly complex. See Figure 4. 4 The centralized approach to managing multiple user views 1 to 3. View integration approach Requirements for each user view remain as separate lists. Data models representing each user view are created and then merged later during the database design stage. The view integration approach involves leaving the requirements for each user view as separate lists of requirements.

We create data models representing each user view. A data model that represents a single user view is called a local logical data model. We then merge the local data models to create a global logical data model representing all user views of the company. A diagram representing the management of user views 1 to 3 using the view integration approach is shown in Figure 4. 5. Generally, this approach is preferred when there are significant differences between user views and the database system is sufficiently complex to justify dividing the work into more manageable parts.

See Figure 4. 5 The view integration approach to managing multiple user views 1 to 3. For some complex database systems it may be appropriate to use a combination of both the centralized and view integration approaches to managing multiple user views. For example, the requirements for two or more users views may be first merged using the centralized approach and then used to create a local logical data model. (Therefore in this situation the local data model represents not just a single user view but the number of user views merged using the centralized approach).

The local data models representing one or more user views are then merged using the view integration approach to form the global logical data model representing all user views. 4. 7Explain why it is necessary to select the target DBMS before beginning the physical database design phase. Database design is made up of two main phases called logical and physical design.

During logical database design, we identify the important objects that need to be represented in the database and the relationships between these objects.

During physical database design, we decide how the logical design is to be physically implemented (as tables) in the target DBMS. Therefore it is necessary to have selected the target DBMS before we are able to proceed to physical database design. See Figure 4. 1 Stages of the database system development lifecycle. 4. 8Discuss the two main activities associated with application design. The database and application design stages are parallel activities of the database system development lifecycle. In most cases, we cannot complete the application design until the design of the database itself has taken place.

On the other hand, the database exists to support the applications, and so there must be a flow of information between application design and database design. The two main activities associated with the application design stage is the design of the user interface and the application programs that use and process the database. We must ensure that all the functionality stated in the requirements specifications is present in the application design for the database system. This involves designing the interaction between the user and the data, which we call transaction design.

In addition to designing how the required functionality is to be achieved, we have to design an appropriate user interface to the database system. 4. 9Describe the potential benefits of developing a prototype database system. The purpose of developing a prototype database system is to allow users to use the prototype to identify the features of the system that work well, or are inadequate, and if possible to suggest improvements or even new features for the database system. In this way, we can greatly clarify the requirements and evaluate the feasibility of a particular system design.

Prototypes should have the major advantage of being relatively inexpensive and quick to build. 4. 10Discuss the main activities associated with the implementation stage. The database implementation is achieved using the Data Definition Language (DDL) of the selected DBMS or a graphical user interface (GUI), which provides the same functionality while hiding the low-level DDL statements. The DDL statements are used to create the database structures and empty database files. Any specified user views are also implemented at this stage.

The application programs are implemented using the preferred third or fourth generation language (3GL or 4GL). Parts of these application programs are the database transactions, which we implement using the Data Manipulation Language (DML) of the target DBMS, possibly embedded within a host programming language, such as Visual Basic (VB), VB. net, Python, Delphi, C, C++, C#, Java, COBOL, Fortran, Ada, or Pascal. We also implement the other components of the application design such as menu screens, data entry forms, and reports.

Again, the target DBMS may have its own fourth generation tools that allow rapid development of applications through the provision of non-procedural query languages, reports generators, forms generators, and application generators. Security and integrity controls for the application are also implemented. Some of these controls are implemented using the DDL, but others may need to be defined outside the DDL using, for example, the supplied DBMS utilities or operating system controls. 4. 11Describe the purpose of the data conversion and loading stage.

This stage is required only when a new database system is replacing an old system. Nowadays, it’s common for a DBMS to have a utility that loads existing files into the new database. The utility usually requires the specification of the source file and the target database, and then automatically converts the data to the required format of the new database files. Where applicable, it may be possible for the developer to convert and use application programs from the old system for use by the new system. 4. 2Explain the purpose of testing the database system. Before going live, the newly developed database system should be thoroughly tested. This is achieved using carefully planned test strategies and realistic data so that the entire testing process is methodically and rigorously carried out. Note that in our definition of testing we have not used the commonly held view that testing is the process of demonstrating that faults are not present. In fact, testing cannot show the absence of faults; it can show only that software faults are present.

If testing is conducted successfully, it will uncover errors in the application programs and possibly the database structure. As a secondary benefit, testing demonstrates that the database and the application programs appear to be working according to their specification and that performance requirements appear to be satisfied. In addition, metrics collected from the testing stage provides a measure of software reliability and software quality. As with database design, the users of the new system should be involved in the testing process.

The ideal situation for system testing is to have a test database on a separate hardware system, but often this is not available. If real data is to be used, it is essential to have backups taken in case of error. Testing should also cover usability of the database system. Ideally, an evaluation should be conducted against a usability specification. Examples of criteria that can be used to conduct the evaluation include (Sommerville, 2000): •Learnability - How long does it take a new user to become productive with the system? Performance - How well does the system response match the user’s work practice? •Robustness - How tolerant is the system of user error? •Recoverability - How good is the system at recovering from user errors? •Adapatability - How closely is the system tied to a single model of work? Some of these criteria may be evaluated in other stages of the lifecycle. After testing is complete, the database system is ready to be ‘ signed off’ and handed over to the users. 4. 13What are the main activities associated with operational maintenance stage.

In this stage, the database system now moves into a maintenance stage, which involves the following activities: •Monitoring the performance of the database system. If the performance falls below an acceptable level, the database may need to be tuned or reorganized. •Maintaining and upgrading the database system (when required). New requirements are incorporated into the database system through the preceding stages of the lifecycle. Chapter 5 Database Administration and Security - Review questions 5. 1Define the purpose and tasks associated with data administration and database administration.

Data administration is the management and control of the corporate data, including database planning, development and maintenance of standards, policies and procedures, and logical database design. Database administration is the management and control of the physical realization of the corporate database system, including physical database design and implementation, setting security and integrity controls, monitoring system performance, and reorganizing the database as necessary. 5. 2Compare and contrast the main tasks carried out by the DA and DBA.

The Data Administrator (DA) and Database Administrator (DBA) are responsible for managing and controlling the activities associated with the corporate data and the corporate database, respectively. The DA is more concerned with the early stages of the lifecycle, from planning through to logical database design. In contrast, the DBA is more concerned with the later stages, from application/physical database design to operational maintenance. Depending on the size and complexity of the organization and/or database system the DA and DBA can be theresponsibilityof one or more people. . 3Explain the purpose and scope of database security. Security considerations do not only apply to the data held in a database. Breaches of security may affect other parts of the system, which may in turn affect the database. Consequently, database security encompasses hardware, software, people, and data. To effectively implement security requires appropriate controls, which are defined in specific mission objectives for the system. This need for security, while often having been neglected or overlooked in the past, is now increasingly recognized by organizations.

The reason for this turn-around is due to the increasing amounts of crucial corporate data being stored on computer and the acceptance that any loss or unavailability of this data could be potentially disastrous. 5. 4List the main types of threat that could affect a database system, and for each, describe the possible outcomes for an organization. Figure 5. 1 A summary of the potential threats to computer systems. 5. 5Explain the following in terms of providing security for a database: authorization; views; backup and recovery; integrity; encryption; RAID. Authorization

Authorization is the granting of a right or privilege that enables a subject to have legitimate access to a system or a system’s object. Authorization controls can be built into the software, and govern not only what database system or object a specified user can access, but also what the user may do with it. The process of authorization involves authentication of a subject requesting access to an object, where ‘ subject’ represents a user or program and ‘ object’ represents a database table, view, procedure, trigger, or any other object that can be created within the database system. Views

A view is a virtual table that does not necessarily exist in the database but can be produced upon request by a particular user, at the time of request. The view mechanism provides a powerful and flexible security mechanism by hiding parts of the database from certain users. The user is not aware of the existence of any columns or rows that are missing from the view. A view can be defined over several tables with a user being granted the appropriate privilege to use it, but not to use the base tables. In this way, using a view is more restrictive than simply having certain privileges granted to a user on the base table(s).

Backup and recovery Backup is the process of periodically taking a copy of the database and log file (and possibly programs) onto offline storage media. A DBMS should provide backup facilities to assist with the recovery of a database following failure. To keep track of database transactions, the DBMS maintains a special file called a log file (or journal) that contains information about all updates to the database. It is always advisable to make backup copies of the database and log file at regular intervals and to ensure that the copies are in a secure location.

In the event of a failure that renders the database unusable, the backup copy and the details captured in the log file are used to restore the database to the latest possible consistent state. Journaling is the process of keeping and maintaining a log file (or journal) of all changes made to the database to enable recovery to be undertaken effectively in the event of a failure. Integrity constraints Contribute to maintaining a secure database system by preventing data from becoming invalid, and hence giving misleading or incorrect results. Encryption

Is the encoding of the data by a special algorithm that renders the data unreadable by any program without the decryption key. If a database system holds particularly sensitive data, it may be deemed necessary to encode it as a precaution against possible external threats or attempts to access it. Some DBMSs provide an encryption facility for this purpose. The DBMS can access the data (after decoding it), although there is degradation in performance because of the time taken to decode it. Encryption also protects data transmitted over communication lines.

There are a number of techniques for encoding data to conceal the information; some are termed irreversible and others reversible. Irreversible techniques, as the name implies, do not permit the original data to be known. However, the data can be used to obtain valid statistical information. Reversible techniques are more commonly used. To transmit data securely over insecure networks requires the use of a cryptosystem, which includes: •an encryption key to encrypt the data (plaintext); •an encryption algorithm that, with the encryption key, transforms the plain text into ciphertext; •a decryption key to decrypt the ciphertext; a decryption algorithm that, with the decryption key, transforms the ciphertext back into plain text. Redundant Array of Independent Disks (RAID) RAID works by having a large disk array comprising an arrangement of several independent disks that are organized to improve reliability and at the same time increase performance. The hardware that the DBMS is running on must be fault-tolerant, meaning that the DBMS should continue to operate even if one of the hardware components fails. This suggests having redundant components that can be seamlessly integrated into the working system whenever there is one or more component failures.

The main hardware components that should be fault-tolerant include disk drives, disk controllers, CPU, power supplies, and cooling fans. Disk drives are the most vulnerable components with the shortest times between failures of any of the hardware components. One solution is the use of Redundant Array of Independent Disks (RAID)technology. RAID works by having a large disk array comprising an arrangement of several independent disks that are organized to improve reliability and at the same time increase performance. Chapter 6 Fact-Finding - Review questions 6. Briefly describe what the process of fact-finding attempts to achieve for a database developer. Fact-finding is the formal process of using techniques such as interviews and questionnaires to collect facts about systems, requirements, and preferences. The database developer uses fact-finding techniques at various stages throughout the database systems lifecycle to capture the necessary facts to build the required database system. The necessary facts cover the business and the users of the database system, including the terminology, problems, opportunities, constraints, requirements, and priorities.

These facts are captured using fact-finding techniques. 6. 2Describe how fact-finding is used throughout the stages of the database system development lifecycle. There are many occasions for fact-finding during the database system development lifecycle. However, fact-finding is particularly crucial to the early stages of the lifecycle, including the database planning, system definition, and requirements collection and analysis stages. It’s during these early stages that the database developer learns about the terminology, problems, opportunities, constraints, requirements, and priorities of the business and the users of the system.

Fact-finding is also used during database design and the later stages of the lifecycle, but to a lesser extent. For example, during physical database design, fact-finding becomes technical as the developer attempts to learn more about the DBMS selected for the database system. Also, during the final stage, operational maintenance, fact-finding is used to determine whether a system requires tuning to improve performance or further developed to include new requirements. 6. 3For each stage of the database system development lifecycle identify examples of the facts captured and the documentation produced. . 4A database developer normally uses several fact-finding techniques during a single database project. The five most commonly used techniques are examining documentation, interviewing, observing the business in operation, conducting research, and using questionnaires. Describe each fact-finding technique and identify the advantages and disadvantages of each. Examining documentation can be useful when you’re trying to gain some insight as to how the need for a database arose.

You may also find that documentation can be helpful to provide information on the business (or part of the business) associated with the problem. If the problem relates to the current system there should be documentation associated with that system. Examining documents, forms, reports, and files associated with the current system, is a good way to quickly gain some understanding of the system. Interviewing is the most commonly used, and normally most useful, fact-finding technique. You caninterviewto collect information from individuals face-to-face.

There can be several objectives to using interviewing such as finding out facts, checking facts, generating user interest and feelings of involvement, identifying requirements, and gathering ideas and opinions. Observationis one of the most effective fact-finding techniques you can use to understand a system. With this technique, you can either par¬ticipate in, or watch a person perform activities to learn about the system. This technique is particularly useful when the validity of data collected through other methods is in question or when the complexity of certain aspects of the system prevents a clear explanation by the end-users.

A useful fact-finding technique is to research the application and prob¬lem. Computer trade journals, reference books, and the Internet are good sources of information. They can provide you with information on how others have solved similar prob¬lems, plus you can learn whether or not software packages exist to solve your problem. Another fact-finding technique is to conduct surveys through questionnaires. Questionnaires are special-purpose documents that allow you to gather facts from a large number of people while maintaining some control over their responses.

When dealing with a large audience, no other fact-finding technique can tabulate the same facts as efficiently. 6. 5Describe the purpose of defining a mission statement and mission objectives for a database system. The mission statement defines the major aims of the database system. Those driving the database project within the business (such as the Director and/or owner) normally define the mission statement. A mission statement helps to clarify the purpose of the database project and provides a clearer path towards the efficient and effective creation of the required database system.

Once the mission statement is defined, the next activity involves identifying the mission objectives. Each mission objective should identify a particular task that the database must support. The assumption is that if the database supports the mission objectives then the mission statement should be met. The mission statement and objectives may be accompanied with additional information that specifies, in general terms, the work to be done, the resources with which to do it, and themoneyto pay for it all. 6. 6What is the purpose of the systems definition stage?

The purpose of the system definition stage is to identify the scope and boundary of the database system and its major user views. Defining the scope and boundary of the database system helps to identify the main types of data mentioned in the interviews and a rough guide as to how this data is related. A user view represents the requirements that should be supported by a database system as defined by a particular job role (such as Manager or Assistant) or business application area (such as video rentals or stock control). 6. How do the contents of a users' requirements specification differ from a systems specification? There are two main documents created during the requirements collection and analysis stage, namely the users' requirements specification and the systems specification. The users' requirements specification describes in detail the data to be held in the database and how the data is to be used. The systems specification describes any features to be included in the database system such as the required performance and the levels of security. 6. Describe one approach to deciding whether to use centralized, view integration, or a combination of both when developing a database system for multiple user views. One way to help you make a decision whether to use the centralized, view integration, or a combination of both approaches to manage multiple user views is to examine the overlap in terms of the data used between the user views identified during the system definition stage. It's difficult to give precise rules as to when it's appropriate to use the centralized or view integration approaches.

As the database developer, you should base your decision on an assessment of the complexity of the database system and the degree of overlap between the various user views. However, whether you use the centralized or view integration approach or a mixture of both to build the underlying database, ultimately you need to create the original user views for the working database system. Chapter 7 Entity-Relationship Modeling - Review questions 7. 1Describe what entities represent in an ER model and provide examples of entities with a physical or conceptual existence.

Entity is a set of objects with the same properties, which are identified by a user or company as having an independent existence. Each object, which should be uniquely identifiable within the set, is called an entity occurrence. An entity has an independent existence and can represent objects with a physical (or ‘ real’) existence or objects with a conceptual (or ‘ abstract’) existence. 7. 2Describe what relationships represent in an ER model and provide examples of unary, binary, and ternary relationships.

Relationship is a set of meaningful associations among entities. As with entities, each association should be uniquely identifiable within the set. A uniquely identifiable association is called a relationship occurrence. Each relationship is given a name that describes its function. For example, the Actor entity is associated with the Role entity through a relationship called Plays, and the Role entity is associated with the Video entity through a relationship called Features. The entities involved in a particular relationship are referred to as participants.

The number of participants in a relationship is called the degree and indicates the number of entities involved in a relationship. A relationship of degree one is called unary, which is commonly referred to as a recursive relationship. A unary relationship describes a relationship where the same entity participates more than once in different roles. An example of a unary relationship is Supervises, which represents an association of staff with a supervisor where the supervisor is also a member of staff.

In other words, the Staff entity participates twice in the Supervises relationship; the first participation as a supervisor, and the second participation as a member of staff who is supervised (supervisee). See Figure 7. 5 for a diagrammatic representation of the Supervises relationship. A relationship of degree two is called binary. A relationship of a degree higher than binary is called a complex relationship. A relationship of degree three is called ternary. An example of a ternary relationship is Registers with three participating entities, namely Branch, Staff, and Member.

The purpose of this relationship is to represent the situation where a member of staff registers a member at a particular branch, allowing for members to register at more than one branch, and members of staff to move between branches. Figure 7. 4 Example of a ternary relationship called Registers. 7. 3Describe what attributes represent in an ER model and provide examples of simple, composite, single-value, multi-value, and derived attributes. An attribute is a property of an entity or a relationship. Attributes represent what we want to know about entities.

For example, a Video entity may be described by the catalogNo, title, category, dailyRental, and price attributes. These attributes hold values that describe each video occurrence, and represent the main source of data stored in the database. Simple attribute is an attribute composed of a single component. Simple attributes cannot be further subdivided. Examples of simple attributes include the category and price attributes for a video. Composite attribute is an attribute composed of multiple components. Composite attributes can be further divided to yield smaller components with an independent existence.

For example, the name attribute of the Member entity with the value ‘ Don Nelson’ can be subdivided into fName (‘ Don’) and lName (‘ Nelson’). Single-valued attribute is an attribute that holds a single value for an entity occurrence. The majority of attributes are single-valued for a particular entity. For example, each occurrence of the Video entity has a single-value for the catalogNo attribute (for example, 207132), and therefore the catalogNo attribute is referred to as being single-valued. Multi-valued attribute is an attribute that holds multiple values for an entity occurrence.

Some attributes have multiple values for a particular entity. For example, each occurrence of the Video entity may have multiple values for the category attribute (for example, ‘ Children’ and ‘ Comedy’), and therefore the category attribute in this case would be multi-valued. A multi-valued attribute may have a set of values with specified lower and upper limits. For example, the category attribute may have between one and three values. Derived attribute is an attribute that represents a value that is derivable from the value of a related attribute, or set of attributes, not necessarily in the same entity.

Some attributes may be related for a particular entity. For example, the age of a member of staff (age) is derivable from the date of birth (DOB) attribute, and therefore the age and DOB attributes are related. We refer to the age attribute as a derived attribute, the value of which is derived from the DOB attribute. 7. 4Describe what multiplicity represents for a relationship. Multiplicity is the number of occurrences of one entity that may relate to a single occurrence of an associated entity. 7. 5What are business rules and how does multiplicity model these constraints?

Multiplicity constrains the number of entity occurrences that relate to other entity occurrences through a particular relationship. Multiplicity is a representation of the policies established by the user or company, and is referred to as a business rule. Ensuring that all appropriate business rules are identified and represented is an important part of modeling a company. The multiplicity for a binary relationship is generally referred to as one-to-one (1: 1), one-to-many (1:*), or many-to-many (*:*). Examples of three types of relationships include: •A member of staff manages a branch. A branch has members of staff. •Actors play in videos. 7. 6How does multiplicity represent both the cardinality and the participation constraints on a relationship? Multiplicity actually consists of two separate constraints known as cardinality and participation. Cardinality describes the number of possible relationships for each participating entity. Participation determines whether all or only some entity occurrences participate in a relationship. The cardinality of a binary relationship is what we have been referring to as one-to-one, one-to-many, and many-to-many.

A participation constraint represents whether all entity occurrences are involved in a particular relationship (mandatory participation) or only some (optional participation). The cardinality and participation constraints for the Staff Manages Branch relationship are shown in Figure 7. 11. 7. 7Provide an example of a relationship with attributes. An example of a relationship with an attribute is the relationship called PlaysIn, which associates the Actor and Video entities. We may wish to record the character played by an actor in a given video.

This information is associated with the PlaysIn relationship rather than the Actor or Video entities. We create an attribute called character to store this information and assign it to the PlaysIn relationship, as illustrated in Figure 7. 12. Note, in this figure the character attribute is shown using the symbol for an entity; however, to distinguish between a relationship with an attribute and an entity, the rectangle representing the attribute is associated with the relationship using a dashed line. Figure 7. 12 A relationship called PlaysIn with an attribute called character. . 8Describe how strong and weak entities differ and provide an example of each. We can classify entities as being either strong or weak. A strong entity is not dependent on the existence of another entity for its primary key. A weak entity is partially or wholly dependent on the existence of another entity, or entities, for its primary key. For example, as we can distinguish one actor from all other actors and one video from all other videos without the existence of any other entity, Actor and Video are referred to as being strong entities.

In other words, the Actor and Video entities are strong because they have their own primary keys. An example of a weak entity called Role, which represents characters played by actors in videos. If we are unable to uniquely identify one Role entity occurrence from another without the existence of the Actor and Video entities, then Role is referred to as being a weak entity. In other words, the Role entity is weak because it has no primary key of its own. Figure 7. 6 Diagrammatic representation of attributes for the Video, Role, and Actor entities.

Strong entities are sometimes referred to as parent, owner, or dominant entities and weak entities as child, dependent, or subordinate entities. 7. 9Describe how fan and chasm traps can occur in an ER model and how they can be resolved. Fan and chasm traps are two types of connection traps that can occur in ER models. The traps normally occur due to a misinterpretation of the meaning of certain relationships. In general, to identify connection traps we must ensure that the meaning of a relationship (and the business rule that it represents) is fully understood and clearly defined.

If we don’t understand the relationships we may create a model that is not a true representation of the ‘ real world’. A fan trap may occur when two entities have a 1:* relationship that fan out from a third entity, but the two entities should have a direct relationship between them to provide the necessary information. A fan trap may be resolved through the addition of a direct relationship between the two entities that were originally separated by the third entity. A chasm trap may occur when an ER model suggests the existence of a relationship between entities, but the pathway does not exist between certain entity occurrences.

More specifically, a chasm trap may occur where there is a relationship with optional participation that forms part of the pathway between the entities that are related. Again, a chasm trap may be resolved by the addition of a direct relationship between the two entities that were originally related through a pathway that included optional participation. Chapter 8 Normalization – Review questions 8. 1Discuss how normalization may be used in database design. Normalization can be used in database design in two ways: the first is to use ormalization as a bottom-up approach to database design; the second is to use normalization in conjunction with ER modeling. Using normalization as a bottom-up approach involves analyzing the associations between attributes and, based on this analysis, grouping the attributes together to form tables that represent entities and relationships. However, this approach becomes difficult with a large number of attributes, where it’s difficult to establish all the important associations between the attributes. Alternatively, you can use a top-down approach to database design.

In this approach, we use ER modeling to create a data model that represents the main entities and relationships. We then translate the ER model into a set of tables that represents this data. It’s at this point that we use normalization to check whether the tables are well designed. 8. 2Describe the types of update anomalies that may occur on a table that has redundant data. Tables that have redundant data may have problems called update anomalies, which are classified as insertion, deletion, or modification anomalies. See Figure 8. 2 for an example of a table with redundant data called StaffBranch.

There are two main types of insertion anomalies, which we illustrate using this table. Insertion anomalies (1)To insert the details of a new member of staff located at a given branch into the StaffBranch table, we must also enter the correct details for that branch. For example, to insert the details of a new member of staff at branch B002, we must enter the correct details of branch B002 so that the branch details are consistent with values for branch B002 in other records of the StaffBranch table. The data shown in the StaffBranch table is also shown in the Staff and Branch tables shown in Figure 8. 1.

These tables do have redundant data and do not suffer from this potential inconsistency, because for each staff member we only enter the appropriate branch number into the Staff table. In addition, the details of branch B002 are recorded only once in the database as a single record in the Branch table. (2)To insert details of a new branch that currently has no members of staff into the StaffBranch table, it’s necessary to enter nulls into the staff-related columns, such as staffNo. However, as staffNo is the primary key for the StaffBranch table, attempting to enter nulls for staffNo violates entity integrity, and is not allowed.

The design of the tables shown in Figure 8. 1 avoids this problem because new branch details are entered into the Branch table separately from the staff details. The details of staff ultimately located at a new branch can be entered into the Staff table at a later date. Deletion anomalies If we delete a record from the StaffBranch table that represents the last member of staff located at a branch, the details about that branch are also lost from the database. For example, if we delete the record for staff Art Peters (S0415) from the StaffBranch table, the details relating to branch B003 are lost from the database.

The design of the tables in Figure 8. 1 avoids this problem because branch records are stored separately from staff records and only the column branchNo relates the two tables. If we delete the record for staff Art Peters (S0415) from the Staff table, the details on branch B003 in the Branch table remain unaffected. Modification anomalies If we want to change the value of one of the columns of a particular branch in the StaffBranch table, for example the telephone number for branch B001, we must update the records of all staff located at that branch.

If this modification is not carried out on all the appropriate records of the StaffBranch table, the database will become inconsistent. In this example, branch B001 would have different telephone numbers in different staff records. The above examples illustrate that the Staff and Branch tables of Figure 8. 1 have more desirable properties than the StaffBranch table of Figure 8. 2. In the following sections, we examine how normal forms can be used to formalize the identification of tables that have desirable properties from those that may potentially suffer from update anomalies. . 3Describe the characteristics of a table that violates first normal form (1NF) and then describe how such a table is converted to 1NF. The rule for first normal form (1NF) is a table in which the intersection of every column and record contains only one value. In other words a table that contains more than one atomic value in the intersection of one or more column for one or more records is not in 1NF. The non 1NF table can be converted to 1NF by restructuring original table by removing the column with the multi-values along with a copy of the primary key to create a new table.

See Figure 8. 4 for an example of this approach. The advantage of this approach is that the resultant tables may be in normal forms later that 1NF. 8. 4What is the minimal normal form that a relation must satisfy? Provide a definition for this normal form. Only first normal form (1NF) is critical in creating appropriate tables for relational databases. All the subsequent normal forms are optional. However, to avoid the update anomalies discussed in Section 8. 2, it’s normally recommended that you proceed to third normal form (3NF).

First normal form (1NF) is a table in which the intersection of every column and record contains only one value. 8. 5Describe an approach to converting a first normal form (1NF) table to second normal form (2NF) table(s). Second normal form applies only to tables with composite primary keys, that is, tables with a primary key composed of two or more columns. A 1NF table with a single column primary key is automatically in at least 2NF. A second normal form (2NF) is a table that is already in 1NF and in which the values in each non-primary-key column can be worked out from the values in all the columns that makes up the primary key.

A table in 1NF can be converted into 2NF by removing the columns that can be worked out from only part of the primary key. These columns are placed in a new table along with a copy of the part of the primary key that they can be worked out from. 8. 6Describe the characteristics of a table in second normal form (2NF). Second normal form (2NF) is a table that is already in 1NF and in which the values in each non-primary-key column can only be worked out from the values in all the columns that make up the primary key. 8. Describe what is meant by full functional dependency and describe how this type of dependency relates to 2NF. Provide an example to illustrate your answer. The formal definition of second normal form (2NF) is a table that is in first normal form and every non-primary-key column is fully functionally dependent on the primary key. Full functional dependency indicates that if A and B are columns of a table, B is fully functionally dependent on A, if B is not dependent on any subset of A. If B is dependent on a subset of A, this is referred to as a partial dependency.

If a partial dependency exists on the primary key, the table is not in 2NF. The partial dependency must be removed for a table to achieve 2NF. See Section 8. 4 for an example. 8. 8Describe the characteristics of a table in third normal form (3NF). Third normal form (3NF) is a table that is already in 1NF and 2NF, and in which the values in all non-primary-key columns can be worked out from only the primary key (or candidate key) column(s) and no other columns. 8. 9Describe what is meant by transitive dependency and describe how this type of dependency relates to 3NF. Provide an example to illustrate your answer.

The formal definition for third normal form (3NF) is a table that is in first and second normal forms and in which no non-primary-key column is transitively dependent on the primary key. Transitive dependency is a type of functional dependency that occurs when a particular type of relationship holds between columns of a table. For example, consider a table with columns A, B, and C. If B is functionally dependent on A (A ? B) and C is functionally dependent on B (B ? C), then C is transitively dependent on A via B (provided that A is not functionally dependent on B or C).

If a transitive dependency exists on the primary key, the table is not in 3NF. The transitive dependency must be removed for a table to achieve 3NF. See Section 8. 5 for an example. Chapter 9 Logical Database Design – Step 1- Review questions 9. 1Describe the purpose of a design methodology. A design methodology is a structured approach that uses procedures, techniques, tools, and documentation aids to support and facilitate the process of design. 9. 2Describe the main phases involved in database design. Database design is made up of two main phases: logical and physical database design.

Logical database design is the process of constructing a model of the data used in a company based on a specific data model, but independent of a particular DBMS and other physical considerations. In the logical database design phase we build the logical representation of the database, which includes identification of the important entities and relationships, and then translate this representation to a set of tables. The logical data model is a source of information for the physical design phase, providing the physical database designer with a vehicle for making tradeoffs that are very important to the design of an efficient database.

Physical database design is the process of producing a description of the implementation of the database on secondary storage; it describes the base tables, file organizations, and indexes used to achieve efficient access to the data, and any associated integrity constraints and security restrictions. In the physical database design phase we decide how the logical design is to be physically implemented in the target relational DBMS. This phase allows the designer to make decisions on how the database is to be implemented. Therefore, physical design is tailored to a specific DBMS. . 3Identify important factors in the success of database design. The following are important factors to the success of database design: •Work interactively with the users as much as possible. •Follow a structured methodology throughout the data modeling process. •Employ a data-driven approach. •Incorporate structural and integrity considerations into the data models. •Use normalization and transaction validation techniques in the methodology. •Use diagrams to represent as much of the data models as possible. •Use a database design language (DBDL). Build a data dictionary to supplement the data model diagrams. •Be willing to repeat steps. 9. 4Discuss the important role played by users in the process of database design. Users play an essential role in confirming that the logical database design is meeting their requirements. Logical database design is made up of two steps and at the end of each step (Steps 1. 9 and 2. 5) users are required to review the design and provide feedback to the designer. Once the logical database design has been ‘ signed off’ by the users the designer can continue to the physical d