Database management system scanned with camscanneruniquely identifies the types of the relation. This is called the primary key of the relation. In the given tables: so, pm and(so, pm) are primary key of the tables supplier, parts and up respectively. Advantages: a) user friendly representation. Given a query: . Find st for suppliers who supply pts i. . . . Find it for parts supplied by a+ is". Both the queries incorporate similar search types and are symmetric. In the relational model, symmetric query solving is symmetric and poses no problem. Ii) storage operations(insertion, deletion, update on) are simple to make. . Hierarchical model: pants nut red of folk tape pm. Suppliers a. Rage ram mumbai 30081. . . . . Of psi. . . Of is a. A. Dharma collect a of a> at incorporated correlations hip maintenance. (we might have placed supplier at the root instead of"parts") entities are represented as free structures. The structure implies the relationship. :) in this particular relation, "part"is superior to"supplier"for each part, there may be more than one supplier record occurrences. Each supplies record also contains the shipment scanned with camscannerquantity. The association is represented by the parent child relationship, hence the name. Iii) in general, the roof may have any number of depend once. Each of these may again have any number of depend dances and so on. In> asymmetric structure not convenient for use advantages and disadvantages: a) symmetric query solving is not symmetric. A) the hierarchical model possesses certain undesirable properties with respect to storage operations. Insert: to introduce a new supplier, dummy parts are to be introduced unless the supplier supplies some part. Delete: deleting a shipment information is to delete the supplier record, implying that we may lose all information about the supplier, if it happens to be the only shipment for the supplier. Update: problems due to redundant occurrences. Any change, in say city of a supplier, requires searching through"the entire database for all such supplier records. . A) the advantage is the naturalness, as natural processes are inherently hierarchical. . Network model: an extension of the hierarchical model. Sib. Rage ram mumbai a. A. Dharma kolkatak3002001400300pl. . . . Pm pm. . . Scanned with camscannerremark: data is represented as records and links. Entity oct us as records and associations as links. More general than hiera rchical structure connector chain represents shipment quantities. No concepts of superiors and dependence. Each connector occurrence links only one supplies and only one part. Advantages and disadvantages: a) symmetric query solving is symmetric. Of) storage operations are simple. Of a) disadvantage is the complexity of links. Relational database deign: relational. . The output of a data real designhierarchialbase design is the uverwoldnetworkview. Sit is the responsibility of the aba to design the database, a assigning the related data items of the database to columns of tables(with respect to a relational model) , in a manner that preserves durable properties. The database designer has to consider many issues at the same time. The final output of the logical database design process is the were schema, since the users schema rep regents the database designers solution. The users schemata are usually difficult to understand and change. The designer is construed by the limited data structure types supported by the database system(since the designer has to keep in mind the access paths. The search strategies are dependant on scanned with cams annette conceptual schema.) the designer may have to consider the access paths of the record, i. A. , how to access a particular record type. The designer may have to consider how to make the set several and updating more efficient. There are two common technologies entity relationship approach and normalization approach towards database design. However, it turns out that the relational design bound on either approach transform into relational form having nearly identical results, and in fact, the two approaches reinforce each other. Or approach: the key idea to the or approach is to concentrate on the conceptual schema. At this stage, the designer should view the data from the point of view of the whole enterprise. "this description is called enter use conceptual schema or enterprise schema. This should be a pure representation of the real world and independent of storage and efficiency considerations. The de sign process can be viewed as a a phase process. Design enterprise schema. . . Translate enterprise schema to user schema for the data blue system. . Advantages of the a phase approach, a) the database design process becomes simpler and better organized. ") the enterprise schema is easier to design than the final schema, since, it need not be a restricted by the ca pabilities of the data scanned with capstan erase system, and is independent of storage and efficient considerations. I) enterprise schema is more stable than the users schema. If one wants to change from one database system to another, one would probably have to change the were schema, but not the enterprise schema in the enterprise schema represented by the or diagram is more easily understood by non exp(electronic data processing) people. Or concepts: the or approach defines a number of data classification objects. Three fundamental data classification objects are: . Entities. Relationships. . Attribute entity: an entity is a collection of distinguishable real world objects, having common properties, and is of interest to the enterprise. Is represented as rectangular boxes in ans or diagram. Employeeprojectdepartmentan entity is usually mapped to an actual table, and each row of the table corresponds to one of the distinguishable objects that make up the entity, called an entity occurrence/ entity instance. Remark: there are many things in the real world, and only some of them are of interest to the enterprise. Its the responsibility of the database designer to select the entities, which are important. Scanned with camscannerremark: choice of entities is a key step to database design. Relationship given an ordered set of entities a, , a. . . . . . Pm. (may not be distinct) a relationship a defines a smile of correspondence between the instances of these entities. Specifically, a represents a set of in tuples"(new. . . . "pm) lei a i, i seem? ? which is basically a subset of the cartesian product of the entities: six sex. . . Pm(rather the entity instances) . A particular occurrence of a relationship, corresponding to a tuple of entity instances(ellen) . . "pm) , where oil i and isis, is called a relationship instance/ relationship occurrence. A is the degree of the relationship. Is represented as diamond shaped bosses in an or diagram. Employee works projects belongs binary relationship assignedtotodepartmentstypes of relationships: a) one to one: for each entity instance in either entity, there is most one associated member of the other entity. A. A. Head of(employee, deportment) of minimum cardinality(a, a) maximum cardinality(a, a) minimum cardinality(of, a) maximum cardinality(for) scanned with cam scanner) many to one: a relationship is many to one, from entity to entity of, if one entity instance in of is associated to a or more entity instances in a, but each instance in a is associated with at most one entity instance in of. A. A: belongs to(employee, department) a minimum cardinality(a, a) maximum cardinality(a, a) minimum cardinality(of, a) masc mum cardinality(for) a. A) many to many: a relationship is many to many from entity to entity of, it one entity occurrence in of is associated with any number of entity occurrences of a, and each entity occurrence in is associated with any number of entity occurrences off. A. A: works on(employee, project) of minimum cardinality(a, a) maximum cardinality(a, a) minimum cardinality(for) masc item cardinality(for) a. A. Remark: there may be many types of relationships between entities, and some of them may not be of interest to the enter use. The database designer is responsible for the selection of relationships relevant to the enterprise. Female: choice of relationships is a key step to database design. Attributes: an attribute is a data item that describes a scanned with camscannerproperly of an entity or a relationship. Be represented a. Oval shaped boxes in an for diagsiam. Else named sig) pop name to works balemployeeprojectson(dur, add a belongsassignedtotodepartmentsd name(a bare outline of the or diagram of the database) types of attributes: a) simple single valued attributes: simple single. Valued att tributes are called simple attributes a: a name, duration. A) composite attributes: composite attributes can be divided into pants, of: address: street+ state+ pin code. Composite attributes help us to group together related attributes, making the modelling cleaner. A) multi valued attributes: multi valued attributes are those that can take on multiple values for a single entity instance. A. A: phone, hobby. Keys) super key: a super key is a set of one or more attributes, that when taken collectively allows us to identify uniquely an entity instance in an entity. Say if let were absent in the above data a base, then(a name, address) could be treated as a super key. Scanned with camscanneragain(a name, designation, address) is also a super key. Thu clearly, there may exist many super keys; any superuser of super key is also a super key in an entity. A) minimal super key or candidate key: a super key for which no proper subset is a super key is said to be a candidate key. It is possible that several distinct set of attributes could serve as a candidate ke

y. Of: of, (a name, add.) composite) primary key: the pol mary key is a candidate key that is chosen by the database designer as the principal means of identifying entity occurrences, within an entity. Usually, april may key is wed in references from other tables. Seq: it, it, it in employee, project and department respectively. (not considering phone+ as an attribute.) transformation rules: a) each entity in an or diagram is mapped to a single table in a relational database. The table is named after the entity. A columns of the tables represent all the single valued simple attributes that are attached to the entity. A pori many key is selected for the entity. Easily occurrences are mapped to the rows of the table. Of) given an entity a, with primary key attribute a; a multi valued attribute a attached to a in an or diagram, is able of its own. Mapped to a table of its own. The table is named after the plural multi valued attribute(a. A. Phones) . The columns of the new table are named after a(it) and a(phone) , and the rows of the table correspond to(a, a) value pairs representing all pairings of attribute values of a, associated with entity occurrences relative top in a. The primary scanned with capstan jersey attribute for this table is the set of columns in and a. Iii, when two entities and of, take part in a many to one, binary relation a, and the entity of represents the many side, of the relationship, the relational tablet, transformed from entity should include columns constituting the primary keys from the table transformed from a. This is known as a foreign key in remark. This foreign key cannot take on null values. A) when two entities a and of take part in a many to many binary relationship a, the relationship is mapped to a sep in tentative tablet, in the related relational database de sign. The table contains columns for all attributes in the pro many keys of both tables transformed from entities a and of, and this set of columns for me the pori many key for to. It also contains columns for all attributes attached to the on elation ship(here, primary keys of either tables in the relationship are not attached to the transformed tables of the opposite entities, at this leads to data redundancy. The phenomenon of treating a relationship like an entity, as described above, is termed aggregation. Refinement allows answering queries pertaining to cross entity references in a relational database. Re finement biking forth the tome essence of relationships in a relational data[base design.) refined or diagram, as compared to the bone outline of the or diagram: scanned with camscannerdatabase: a database is a collection of stored operational data used by the application systems of some particular enterprise(a. Co. Eagles) database components: the components of a database are: database: a large collection of data, stored in second storage. Application programs: run against this data, operating on in all usual ways. On line were: interacting through terminals, performing a functions(mainly retrieval) direct application programs were(less interactive) (greater interaction) . Database/ data storage. (overlapped units: portions of database peeved by single very/ groups bourdon requirement/ access sights etc.) . Interaction with a database, primarily includes retrieval, insertion, deletion, update on(via direct user application programs) the database is integrated, so that the database contains data for many were, and not just one. This implies: scanned with cam scanner a named sights work a name add. Sal. A to employee reworks projects phone dump belongs assigned to(many to(many to. Foreign many) aggregation need(many to. One) has a separate departments table mapped to it with attributes a) a name it and phone. Definitions: a) weak entity: a weak entity is an entity whose occurrence are dependant for their existence through a relationship a on the occurrence of another entity called strong entity. There maybe occasional cases in which the entity occurrence of an entity are not distinguished by their attributes but rather by their relationships to entities of another type. These entities are called weak entities. Whereas, the entities whose entity occurrences can be distinguished by its keys are called strong entities. Of: personnel(strong entity) is a(weak entity) pilot scanned with cam scanner) any one user will be concerned with just a small portion of ii) different users portion, will overlap in various ways, i. A. , individual pieces of data may be shared by different very. Online users: end users: most interactive. . Casual users: users accessing the database with some commercial query language. . Naive users: users accessing the database through venues. Application programmers: write the menu application used by naive users. The program must foresee the needs of the users, and be able to pose queues dur ing execution, to retrieve desired information from the database. Database administrator(aba) : the aba is a team of computer professionals responsible for the design and maintenance of the database. An enterprise is a reasonably large organization a manufacturing company, university, hospital, government organization that maintain data that can form a database. Operational data: it does not include any purely transient data(ill, work queue etc.) . Transactions may cause a change to the operational data, but are not a part of the database. (transactions in hotel management systems, airline reservation systems etc. May be required to be store scanned with cam scanners operational data, as customers may request a detailed transaction listing.) associations or relationships between data, must also be stored as part of operational data. A. A: for a manufacturing company details of projects handled, use of parts supplied by suppliers warehouse locations employees: each being a distinguishable entity, has relationships/ associations with one another. Centralized database : so) centralized control of operational data. Of) the aba has the central responsibility of the operational data(taking care of critical section issues) . Tasks of a aba: a) deriding information content. Of) designing the database. Organize views, restrictions etc. Iii) recovery mechanism(frequency of backup issues) and backups. In) authorization checks. . A) monitor and upgrade performance issues. Advantages/ implications of a dams: a) the amount of redundancy in the stored data can be red. Used. Independent applications may have their own pour defiles leading to redundancy and wastage of storage space. With centralized control, this redundancy can be reduced. Of) inconsistency: the problem of inconsistency can also be scanned with camscanneravoided to some extent. This property directly follows from redundancy removal, implying, two entries of the same information may lead to inconsistency if one is updated. Remark: sometimes, there are technical reasons for maintaining several distinct copies of the same data. Any such ready? nancy should be carefully controlled. Of a) data hasting: data can be shared, i. A. , not only the existing applications can share the data in the database, but also, new applications can be developed to operate against the same shared data. A) standards: industry standards con be enforced and maintained by the aba. This simplifies problems of maintenance and data interchange between installations. A> security: security a instructions can be applied, the duncan ensure that the only means of access to the database is through proper channels and hence can define authorization checks to be carried out, when access to sensitive data is attempted. This also implies piracy across various user dep"apartments. I) integrity: integrity can be maintained. The data value stored in the database, must satisfy certain types of con st ency constraints. The problem of integrity is the problem of ensuring that the data in the database is accurate. Vii) ease of application development: cost and time for developing new applications are reduced. Studies show that, a programme can develop an application a a times faster. The reason being that the programmer is free from designing, building, and main staining master file scanned with capstan merrill) data independence: separation of data from the paper cation program environment organization of data can change and evolve, without any change in the application program. . A: field size change, file organization change etc. It is amazon objective of database systems and may be def ned as the immunity of applications to change in the storage structure and access strategy, i. A. , the applications once ned do not depend on any particular storage structure or access strategy. May be viewed as a a stage independence. Logical data independence: capacity to change the conceptual schema, without having to change external schemes or application programs. . Physical data independence: capacity to change the internal schema, without having to change the conceptual schema. Schema of a database, stands for the description of a data base. Database architecture: a. Schema conch lecture(anti/ spare conch lecture) : (external ext. View, set. View get view level) external conceptual mapping conceptual(an abstraction: pertains to data as viewed by the no schema/ level conceptual internal and not the actual storage) mapping: given by underlined dam internal(focuses on the actual storage structure. Schema/ level of data, not viewed by the do data as in the secondary storage) scanned with camscannerinternal level. This level has an internal schema, which defines the physical storage structure of the dams, in , how the data base is actually stored(depends on the data model of the bus) . It de tubes the complete details of data storage and access paths for the database. Conceptual level: this level is defined by a conceptual schema through the data definition language(day) of the bus package used. This is done by the aba, who decides what information is to be kept in the database. This is a representation of the entire information content in the database, in a form that is somewhat abstract, in comparison to the way in which the data is actually(physically) stored. (leg : physically a a tree, but conceptually a table) it maybe completely different from the way data is viewed by the user. It describes what data are actually stored in the data base, and the relationships that exist among the data . External level: this level is defined by a number of ext or email schemes or views. Most database users will not be concerned with the entire database, but need just a part of it. Thus, there are many external views of it, which basically consists of definitions of various external record types in that view. Remark: the conceptual schema is intended to include additional features like author station checks and validation procedures. The does is a software that: scanned with cam scanner) allows the aba to define the conceptual and external model through the day. In most cases, the physical model definition is part of the package. ") allows users to manipulate data through data manipulation language(del) commands and routines. In) handles all accesses to the database, i. A. , controls the overall operation of the database. Data definition language: a high level non procedural las quake. It is a notation for describing the entities and relationships among the data entities, in terms of a particular data model. It is used to: a) express the design of the database. Of) modify the design. I) describe in abstract terms what the physical lay out of the database should be. Specify domain constraints, referential integrity, assertions, authorization data manipulation language: used to manipulate data, i. Perform storage(insert, deletion, rets ere etc.) operations. Remark: the del and the del depend on the dams package which in turn depends on the data model. Data models: in a centralized database, there are type of data models which form the heart of a database: a) relational of) hear archival chi network, . Relational model: scanned with camscannersupplier: (entity: provides complete information) is name tyson. Rage ram mumbai a. A. Dharma pol ka a. Table synonymous with relation parts: (entity) . Record synonym a namecolourcwf. City. Mouse with instancednutred12 polka for tuple. . . A bott green17mumbaipsscrewblue17delhisp: (a table that symbolizes a relations post more than an entity) a. Of asp of the statement: supplier no. Of of names. Page web. Rage ram stays in mumbai: depictss2p300a relationship which is best explained in the supplier table. A) entities and also citations are viewed as tables core relationships) of) most convenient form for the users. In) for every tuple, a relation exists between every attribute of the table, and the same relation holds for every tuple within a relation. And thus the name relational model. Within a given relation, there is one attribute(may be a composite attribute of a or more attributes) with values that scanned with cam scanner. In the example, a pilot is a specialization of personnel, or a personnel is a generalization of pilot. The pilot entity may not have a key, but can be identified by the personnel identified. Is a is a weak relationship) existence dependant entity: the existence of an entity occurrence may sometimes depend on the assistance of another entity occurrence in another entity. A. A. Employee parent of children(existence dependant) in this particular case, the existence of children entity depends on the existence of the associated employee. It an employee leaves the company, the database shall not keep track of the children, thus"children"is an existence dependant entity. This is also a weak entity, though the weak entity children may have a key attribute children. The relationship parent of is also a weak relationship(many to many) . Its possible that the existence dependant relationship is a many to many mapping. A. A: if the father leaves the company, the children entity occurrence may still assist of their mother is an employee of the company. A) id dependency: if an entity cannot be uniquely identified by its own attribute, and has to be identified by its svelte scanned with camscanneronship with other entity(a) , then we say that it has an id dependency on other entities. A. A. A street is unique only within a city, a city is unique only within a state and a state is unique only within a country. Countryconsistsofinorder to uniquely ideas testify the address of a local i don, we have to specify the names of city, state and country, in addition to the name of the street. Remark: an id dependant cities by is automatically exist dance constrained, but an existence constrained is botha necessary an id dependant since, the existence dependent entity can still be uniquely identified by instr betsey. I remark: choice of attributes is a key step to database design. Scanned with camscannerdependancies and normal, form: the goal in relational design is to choose relations that remain consistent and have minimum inconsistency, such relations on said to be in the normal. Form, in a normalized situation, of every now and column position in the table, there exists precisely one value, and never a set of values, i. A. , in a normalized net"ion, each of the underlined domains contain atomic value only. Functional dependency(for) : given a relation a, the attributes of is functionally dependant on attribute a of a, if each so value in a has associated with it precisely one a value in. Is usually represented as: a note: there may be same a values in different types of a. If a is functionally dependant on a, then for the tuples having same a values, they values must also be same. Considering a relation: first: is status city phat. As is evident from the1820londonp1300table, 20p2200. A status and a city are20p3400functionally dependent onsi20p4200st. Is pm of. A status is functionallyis20p6dependant on a city. Sqpoou'sp300. (so, a) form the keys210400of the table. 83pq200s420londonp22200s420p4200s420p5400scanned with cam scanner diagram: she status typha city| difficulties in storage operations: . Insertion if we were to insert a record for. Of, who current. Fly supplies no part; the operation would not be possible asp cannot hold a null value. . Deletion if we were to delete record of of paris. . . . Rather, if we were to delete the transhipment of of; all record corresponding to of would be deleted. . Update on if we were to updates city of of from london to an francisco; we would have to search through all the records and update accordingly. A very tedious task, prone to inconsistencies. I cause of the above problems: absence of full functional de pendency(fed) full functional dependency(fed) : a tribute a is fed on attorn bute a, it it is cd on a and not cd on any proper subset of the attributes of a. In the above cd diagram, a partial dependency is sieve led. Resolution of the difficulties: projection of first as follows, creating two new relations: second: (a: key) up: ((so, pm) : key) is status a city phsty3120londoncrelationbetween supplies supplier81 300information52topooris1032pi800and ports of. . . A 420londonscanned with camscannercorresponding cd diagram. A status rays phi city second. Now, the previously mentioned storage operations are tendered easier. However, some other difficulties ore exposed: insertion: rome: st. Status of cannot be inserted as a for the record would have null values. . Deletion: say, there exists a record of rome of, if it be deleted, the record rome of stands deleted too. . Update on: due to the a redundancy in a city, update on of a. States is error prone. The above difficulties aube due to the transitivity in the dependencies in second, i. A. , the dependency of supplier status on, though functional is transitive. This transitive to leads to difficulties over storage operations. Thus, we replace second by its projections as follows: so; (so: key) is: (a city: key) sis city city. Statussilondonlondon2032paris10rome5034 london diagram: (the cd of up stands unchanged and is to be included) a city city a status scanned with cam scanner. Transitive dependency, a functional dependency by in a relation a is a transitive dependency if there is a set of attributes a, that is neither a candidate key not a subset of any key of a, and both a a and by hold. First normal form(inf) : a relation is in inf, if all the underlined domains contain atomic values only. Seq: first second normal form(inf) : a relation is in ink, if it is in inf, and every non key attribute is fed on the primary key. A. A: first is not in ink, but second and spare in inf. Third normal form(inf) ; a relation is in inf, if it is in one and every non key attribute is non transitively dependant on the primary key. A. A. So, is and spare in a ned. . Summary of normal forms, and the corresponding normalization technique: normal fortes normalization violin relation should have form new relations no monatomic attrib for each monatomic is, or nested relate alto bute oriented one. Relation. In for relations, where decompose and setup the pori many key con a new situation for a gains multiple attrib cd partial key with states, no non key attorn dependant attribute(a) bute should be de on a relation must exist part of the primary with the original poi key. Mary key and all its fast there should be no decompose and ser up transitive dependency a relation that include of a non key attribute the non key attribute(a) on the primary key, that determine others. . Scanned with camscannerdeterminant: any attribute(a) on which any one ore more attributes are fed is(are) called a determinant, i. A . , if key, a is termed as the determinant. In the relations covered, the determinants are as follows: . Firstsecondspsccshs(a, a) so city city. A city. (so, pm) joyce road normal form(bank) : a normalized relation a, can be said to be in bank if every determinant of a is a candidate key. It is in bond, it must be in inf, but the reverses not always tome. Here, up, so and is are in ben. A: note: any bin cory situation is in bank. Given a relation: supp, with alts bytes: so, a name, a status and a city, where . I) a status and a city are independent) both sit and a name come candidate keys. Thus, the corresponding a diagram is: she status is names city here there exists no transitivity, and the determinants are: so, a name and(so, a name) and they are all candidate keys. Thus, supp is in bank. Here, the determinants are disjoint. Given a relation: sep, with attributes: a, a name, pm, city, scanned with camera operand the candidate keys are: (it, pm) , (pm, a name) . Now as is evident, the candidate keys are not dis joint, and the determinants re: (st, it) , (surname, pm) , st, a name. Here all the determinants are not candidate keys, and thus sep is not in bank. Moreover, a name is a redundant attribute, and up a ion of its values poses difficulties. Thus, resolution of the problem is through the projections: in(so or a name: key) is((so, pm) : key) is names(or name) pm other even a relation: of, with attributes: student, subject, teacher. The scales pertaining to the relation are as follows: a) for each subject, each student of that subject is taught by only one teacher. ") each teacher teaches only one subject. Each subject is taught by several teachers. . . . Sjfstudentsubject teacher here, (student, subject) isanilmathssena candidate key. (student, physics teacher) forms an overlapping sadhu matisse candidate too. Thus, an dead physics"duty diagram would be jastudentteachersubject scanned with camera nyerere, the determinants care: (student, subject) , (student, teacher) , teacher; but since teacher cannot be a candidate key, the situation is not in bank. Here, if the tuple canal physics shay by deleted, then the record physics sara is removed. Thus, if is replaced by its projections: i) st(student, teacher) of) to(teacher, subject) . Giv en a relation: exam, with attributes student, subject, position, and i) no two students obtain the same position in the same subject. Thus, the candidate keys ore(student, subject) and(subject position) . Here, the determinants are(student, subject) and(subject, position) . Now, since all the determinants are candidate keys, "the given relation is in bank. Though the determinants and the candidate keys overlap, the absence of a subset of the candidate keys as a determinant results in none of the previous difficulties writing. . Given a situation: cox course teacher text course teacher testalkndzugernavatheaiwillsonnckorthwinston. Navthedbmssckorthhere, . Every course has a givennavathetext set. Sask orth. Several teachers can teach scanned with cam scanner course. . Assume that no matter who actually teaches a particular course, the same texts are used. Meaning of cry: tox is an all key relation. A tuple< city a> appears in cox. If a can be taught by to who uses a as a reference. Note that in the relation cox. If the tuples< < . To, a, a and< a. , to; a) both appear, then tuples< a, to, in) and(a, to; "> also appear. Clearly, cry contains a great deal of redundancy. This leads to problems in upstate operations. A. A. To add the record that dams uses a new text only, it is necessary to add one tuple for each teach of dams, as cox is an all key relation and is in bank. The lack of dependency of teacher on text is the source of the difficulties. . Improvement. Replacement of cry by its all key bond projection? it(course, teacher) , sex(course, test) extension of of the definition of cd may be extended by: the following representations: a a well defined set shaving multiple value of a for a value of a. In the second representation, the concept of multi valued dependency(mid) is brought forth. Remark: although a given course does not have be a single cor scanned with cam scanner. Responding teacher(teacher is not a on course) , nevertheless, each course has a well defined set of corresponding teachers. This is called an mid. There is an mid of teacher on course and an mid of text on course. Clearly, cd is a special case of mid, and hence, mud is a generalization of a, i. A, an cd is an mid in which the set of dependant values, actually consists of a single value. [cd is a specialization of mid] . Theorem: lossless decomposition: a relation a with attributes (a, a, a) can be lossless decomposed into two projections

: i(a, a) andre(a, a) , if, there is an mid of a on a and
an mid of con a in. Fourth normal form(inf) : a normaliz
ed relation a is said to be in inf, whenever there exists
an mid in a, say of attribute cox is not in inf, but it
and cd are in una. Lion alt bute a, the all attributes of
rare also cd on a] relational model and query language:
the relational data model, represents the database as a c
ollection of tables, and there is a direct correspondence
between the concept of a table and the concept of a math
ematical situation. "formal query language: a query langu
age is a language through which a user requests informati
on from the database. These are typically high level lang
uages compared to stand a red programming languages. Quer
y languages ore classified a scanned with camscannerfollo
ws: procedural: specification of a sequence of operations
. A. A: relational algebra. . No procedural: the sequence
of operations is not specified, but the information deci
ded is a: relational calculus. Relational calculus is sub
divided as follows. Tuple relational calculus: a variable
represents a tuple of a relation. . Domain relational ca
lculus: a variable represents a data item or field or att
ribute(simple single called) commercial query language: t
akes features of both procedural and non procedural query
languages. Leg: butchered avery language(sol) relational
algebra relational algebra, essentially encompasses. A)
fundamental operations that allow construction of. "queri
es of our choice. The fundamental operations are select(a
) , project(it) , cartesian product(a) , union(a) and dif
ference() of) a additional operations that are dependant
on the fundamental operations. The additional operations
are: intersection(a) , theta join. (a) , natural join(a)
and division(+) use of any of the above operations on re
lations, yields a new operation hard fundamental operatio
ns: a) han a scanned with cam scanner. Select: a nary ope
ration(i. A. , it operates on a single relation) resultin
g in a host total subset of that situation. Syntax: a(rel
ation name) predicate select rows of relation name, satis
fying conditions in the predicate. The predicate may comp
rise of relational operators(. > < < < > > > >) , logica
l operators/ connectors(and: a, or: a, not:) or a combin
ation of both. . Project: a nary operation resulting in a
vertical subset of the situation under consideration. Sy
ntax: attrib, a tout, , a tour(relation name) a select co

lumns, corresponding to the attribute list specified, of relation name. . Cartesian product: a binary operation li re, it operates on two relations) resulting in the follow ing: a. A relations a relation a where, if relation. I ha s attributes and to, tuples and relation. A has attribute s and to tuples, a has(man) attributes(common attributes are repeated) and(text,) tuples. . Union: a binary opera tion with the following constraints: a) the two relations , i and re, under consideration must be"compatible, i. A. , should have the same number of attributes. A) the with attributes of i and re, must be defined in the same scan ned with camscannerdomain, i. A. , represent the same val ues, though they may be identified by different attribute names. Syntax: i a re. Results in a situation with tuple s of i and re, no repetitions. . Difference: a binary sit uation resulting in tuples of re that core not present in re. Follows constraints of compatibility and syntax: i r e. (set difference) same domain values. The a fundamental operations/ operators allow us to give a complete defini tion of an expression in relational algebra. Kate, and of be two relational algebra expressions. Then, each of the following are relational algebra expressions. A, be. A, in. A, xes(. . . (a) , where a predicate on attribute(a) of a. . . It(i) , where a a list of some of the attribute s of al. Additional operations. . . The five fundamental operations of relational algebra, a, it, a, , are suffici ent to express any relational algebra query. However, som e common queries are lengthy to express with just these o perations. The four additional operations, each of which can be expressed in terms of the five fundamental operati ons, make the query length smaller in some cases. . . Int ersection: syntax: in a. , results in a situation with tu ples common to both i and re. Air erin re i(i re) are(i r e) in a scanned with camscannerremark: it is possible to write several equivalent situational algebra expressions, that are quite different from one another. . Theta join: a binary operation with syntax: i i re[(six re) natural join: a binary operation with syntax: i it re(six to)) r iver i. A, era. A, a i. A re. A. A. . . Aria re. A where a; (i a to a) represents the common att bute set of the r elations. The common attributes occur only once in the re sultant situation. (i, re> > attribute sets of i and re r espectively) . Division: a binary operation with syntax:

i are to(i) to((tori rare pop(i) or a) re) i re(i", rio a
ttribute sets of i and re respectively) . The constraints
of the operation are if i has(man) attributes andre has
the last attributes, in, the(with attribute of coincides
with the with attribute of re, then the division operatio
n yields a relation(a re+ re) that has matt suburbs. Usua
lly satisfies the"for all"clause in a query. Proof of the
equivalence: clubbing/ concatenating together the in alt
s bytes to a single attribute a and the attributes to a f
or i and i, let, i: re: and then i+ try, a, by a(. : , is
associated with each value of a) la tuple to is in is if
: {a(a) , a(a) are relations, ser; is is a relations) to
is in tors(a) a) for every to in a, there is by in a, [a]
to[a] , [, [re. In on is] scanned with cam scanner. Now,
it tori re, (i) by named a, i. A. , a it in res. (i) , t
hen a re. > (a up) r1xxyxy222 a, cry acc it i re((a up) r
e) . A bra+ re. A relational algebra queries: given a sch
ema, "a camp(name, mini, name, san, date, add, sex, salar
y, superman, a no) foreign key. Dept(name, number, mersin
, mgr tartrate) dept low(a number, location) works on(ass
n, no, hours) project(name, number, plot, drum) dependent
(assn, a name, sex, date, relation) . Ret were the name a
nd address of all employees who work in the"research"depa
rtment. Rich dept(dept) a name research scanned with cams
cannerrsch dept emp< rich depth number in result< to(rich
dept emp) name, name, address. For every project located
in stafford, list the project number, the controlling de
partment number and the last name, address and birth date
of the department manager. Pct stafford{a(project) plot
stafford control depth or stafford depth cum number resul
t< it(control dept damp) name, add, date green san. Find
the names of employees who work on all the project contro
lled by department numbered a. Pacts(no) < to((project))
number cum sons(san, no) < to(works on) assn, posses pac
ts< sons pct result< it(sons pacts a emp) name, name retr
ieve the names of employees who have no dependants in dea
n of(dependent) (san) essays is< to(emp) is scanned with
cams anderson de passes san dep result to name, name(san
dean i emp) list the names of managers who have at least
one dependant. Manager(san) < timers in(dept) san dean(sa
n) < items(dependent) manager dean< manager a san dep res
ult to(manager dean a emp) name, name. Make a list of pro
ject numbers or projects that involve an employee whose l

ast name is smith either as a worker or asa manager of th
e department that controls the project. San smith(assn) <
 of assn name smith(emp)) smith worker toss smith a work
s on) no mgr smith(anus) < its in smith dideptdnumbermgrs
sn less smith manager(no) < it(mgr smith i project) numbe
r result< smith worker a. Smith manager. . Relational cal
culus: a) do relational algebra is a procedural language,
 since we write a relational algebra expression, by provi
ding a sequence of operations that generates the answers
to our query. In relational calculus we give a formal des
cription of the information desired, without specifying h
ow to obtain the information. Relational calculus scanned
 with cam scanners non procedural. The two forums of rela
tional calculus ore: tuple relational calculus a variable
 represents a tuple in relation domain relational calculu
s a variable represents a column/ field in a relation. Tu
ple relational calculus(try) : a general query of try is
of the form st/ a(to) ? , i. A. , we would like to obtain
 all tuples"to", satisfying the predicate"a(to) ". An att
ribute of a typical tuple to, is stated as to. A. "a"bein
g an attribute in to. Formal definition of try: a tuple v
ariable is a free variable, unless quantified by] (there"
exists) or a(for all) , else they are bound variables. An
y numb, or of tuple variables may appear in a formula. A
the formula is built up of atoms. An atom has the followi
ng forums: a) ser, where a is a tuple variable and a is a
 relation(the use of the operator is not permitted) a) a.
 A a a. A, where a and a are tuple variables, be is an at
tribute on which a is defined, a is an attribute on which
 a is a defined, and a is a comparison operator(< . < ()
, > , > >) ; it is required that attributes a and a have
 domains choose members can be compared by of. A) a. A of
, whereas is a tuple variable, in is an attribute on whic
h a is defined, of is a comparison operator, and a is a c
onstant in the domain of attribute a. Formulae are built
up of atoms, using the following tiles: and a. A) an atom
 is a formula. A) if a. Is a formula, then so one: a, and
(a) a) if a, and to are formulae, then so one: a. App, pi
n up. A) if a. (a) is a formula containing a free tuple v
ariables, and is a scanned with camera interrelation, the
n is or(a. (a)) , user(a. (a)) are also formulae. Safe
try: a try expression may generate an infinite relation.
A. A. It/ be(to) ; may not be a finite set, and may conta

in values that do not belong to the database. Thus, such
the expressions ore unsafe. To define a destruction on th
e the expressions, we introduce the concept of domain of
a tuple relational formula a. "the domain of a[dom(a)] i
n the set of all values that appear explicitly in for tha
t appear in one or more relations whose names appearing.
An expression a top(to) ? is safe, if all the following h
olds a) all values that appear in tuples of the expressio
ns, are values from dom(a) . A) for every expression like
is(a. (a)) in a, the sub simulate true, if there is a t
uple. A, with values from dom(a.) , such that a(a) is tr
ue. A) for every expression like is(a. (a)) in a, the su
b simulate true, if a, (a) is toque for every tuples, wit
h values from dom(a) . Safe the expressions are equivalen
t to relational ages expressions. Queries using schema: "
acc". Retrieve the birth date and address of employees na
med john smith. Date, a. Address| emp(+) and to. Name jo
hn and. Mini by and to. Name: smiths. Ret were the name a
nd address of all employees who work for the research dep
artment. Scanned with cam scanner. Now, it tori re, (i) b
y named a, i. A. , a it in res. (i) , then a re. > (a up)
r1xxyxy222 a, cry acc it i re((a up) re) . A bra+ re. A
relational algebra queries: given a schema, "a camp(name,
mini, name, san, date, add, sex, salary, superman, a no)
foreign key. Dept(name, number, mersin, mgr tartrate) de
pt low(a number, location) works on(assn, no, hours) proj
ect(name, number, plot, drum) dependent(assn, a name, sex
, date, relation) . Ret were the name and address of all
employees who work in the"research"department. Rich dept(
dept) a name research scanned with camscannerrsch dept em
p< rich depth number in result< to(rich dept emp) name, n
ame, address. For every project located in stafford, list
the project number, the controlling department number an
d the last name, address and birth date of the department
manager. Pct stafford{a(project) plot stafford control d
epth or stafford depth cum number result< it(control dept
damp) name, add, date green san. Find the names of emplo
yees who work on all the project controlled by department
numbered a. Pacts(no) < to((project)) number cum sons(s
an, no) < to(works on) assn, posses pacts< sons pct resul
t< it(sons pacts a emp) name, name retrieve the names of
employees who have no dependants in dean of(dependent) (s
an) essays is< to(emp) is scanned with cams anderson de p

asses san dep result to name, name(san dean i emp) list the names of managers who have at least one dependant. Manager(san) < timers in(dept) san dean(san) < items(dependent) manager dean< manager a san dep result to(manager dean a emp) name, name. Make a list of project numbers or projects that involve an employee whose last name is smith either as a worker or asa manager of the department that controls the project. San smith(assn) < of assn name smith(emp)) smith worker toss smith a works on) no mgr smith (anus) < its in smith dideptdnumbermgrssn less smith manager(no) < it(mgr smith i project) number result< smith worker a. Smith manager. . Relational calculus: a) do relational algebra is a procedural language, since we write a relational algebra expression, by providing a sequence of operations that generates the answers to our query. In relational calculus we give a formal description of the information desired, without specifying how to obtain the information. Relational calculus scanned with cam scanners non procedural. The two forums of relational calculus ore: tuple relational calculus a variable represents a tuple in relation domain relational calculus a variable represents a column/ field in a relation. Tuple relational calculus(try) : a general query of try is of the form st/ a(to) ? , i. A. , we would like to obtain all tuples"to", satisfying the predicate"a(to) ". An attribute of a typical tuple to, is stated as to. A. "a"being an attribute in to. Formal definition of try: a tuple variable is a free variable, unless quantified by] (there"exists) or a(for all) , else they are bound variables. Any numb, or of tuple variables may appear in a formula. A the formula is built up of atoms. An atom has the following forums: a) ser, where a is a tuple variable and a is a relation(the use of the operator is not permitted) a) a. A a a. A, where a and a are tuple variables, be is an attribute on which a is defined, a is an attribute on which a is a defined, and a is a comparison operator(< . < () , > , > >) ; it is required that attributes a and a have domains choose members can be compared by of. A) a. A of, whereas is a tuple variable, in is an attribute on which a is defined, of is a comparison operator, and a is a constant in the domain of attribute a. Formulae are built up of atoms, using the following tiles: and a. A) an atom is a formula. A) if a. Is a formula, then so one: a, and(a) a) if a, and t o are formulae, then so one: a. App, pin up. A) if a. (a) is a formula containing a free tuple variables, and is a scanned with camera interrelation, then is or(a. (a)) , user(a. (a)) are also formulae. Safe try: a try expression may generate an infinite relation. A. A. It/ be(to) ; may not be a finite set, and may contain values that do not belong to the database. Thus, such the expressions ore unsafe. To define a destruction on the the expressions, we introduce the concept of domain of a tuple relational formula a. "the domain of a[dom(a)] in the set of all values that appear explicitly in for that appear in one or more relations whose names appearing. An expression a top(to) ? is safe, if all the following holds a) all values that appear in tuples of the expressions, are values from dom(a) . A) for every expression like is(a. (a)) in a, the sub simulate true, if there is a tuple. A, with values from dom(a.) , such that a(a) is true. A) for every expression like is(a. (a)) in a, the sub simulate true, if a, (a) is toque for every tuples, with values from dom(a) . Safe the expressions are equivalent to relational ages expressions. Queries using schema: "acc". Retrieve the birth date and address of employees named john smith. Date, a. Address| emp(+) and to. Name john and. Mini by and to. Name: smiths. Ret were the name and address of all employees who work for the research department. Scanned with cam scanner. Or, select a. Name, a. Name rome a, dependant where. Assn: a san and. A name a. Name and. A. Sex a sex. In general, a query written in nested blocks and wing the or"in"operator, can always be expressed as a single block query. . The constructs with"in", "sony", "us", allow us to test a single value against members of a set(an entire set) . Sol also allows< any< any", "> any any, "any", "all"since a select generates a set of tuples, we may at time want to compose sets to determine if one set contains all the members of some other set. Such comparisons are made intel using contains and not contains. A contains a> is xxx not contains a> sex. Retrieve the names of all employees who work on all the projects controlled by department"a", "select name, lnamefromempwhere((select in from works. On here san assn) contains(select number from project where anus of)) scanned with cam scanner exists(a) siphons tome, it attest one there exists in the result of the tuple a. Similarly, we have not exists(a) as

opposed to exists(a) , which such is tore if a is empty.
. An alternative approach to the query pertaining to the
retrieval of names of employees who have dependants of th
e same name and sex. Here, for each tuple of employee, th
e nested query that retrieves all dependant tuples with s
ame san, name and sex as the employee tuple, is evaluated
. Select name, name rome where exists(select from dependa
nt here a. San assn and. Name name and. A. Sex sex) . "if
at least one tuple exists in the result of the nested qu
ery, then i select that employee tuple for output. Retrie
ve the names of employees who have no dependants. Select
name, lnamefromempwhere not exists(select from dependant
here assn san) a. List the names of managers who have at
least one dependant select name, una me rome where exists
(select scanned with capstan perform dependant here assn
san) and exists(select from dept where mgr san san) conta
ins by cd> except a> a a i. A. , set a contains set a"is
logically equivalent to"a except(set difference) a"is emp
ty. . An alternative approach to the query pertaining to
the retries"al of the names of all employees working in a
ll projects control led by departments. Select name, lnam
efromempwhere not exists((select number from project wher
e dom of) except(select poor works nowhere in assn)) . .
Retrieve the sons of all employees who work on project:
a, for select less from works on here no in(a, a, a) scan
ned with cam scanner. Retrieve the names of all employees
who do not have any supervisors. Select name, lnamefrome
mpwheresuper san is null missing/ undefined/ inapplicable
. . Retrieve the last name of each employee and his/ her
super. Visor, while oven ming the attribute names as"emp
name and"supervisor name"respectively. Select a. Name as
emp name, a. Name as supervisor name rome are, emp ass. W
here a. Superman a. San. Retrieve the name and address of
each employee working in the research department. Select
name, name, add no join from(emp join dept on a no numbe
r) operation. Where name: research aggregate functions an
d grouping: there are a number of built in aggregate func
tions like count, sum, max, min, avg etc. . Find the sum
of salaries of all employees, the maximum salary, the min
imum salary and the average salary. Select sum(salary) ,
max(salary) , min(salary) , avg(salary) from emp. Scanned
with capstan nervine the sum of the salaries of the empl
oyees in the reseal a department, as well as the max, min

and arg salary in this department. Select sum(salary) , max(salary) , min(salary) , avg(salary) from emp, depth herein number and name: research. Find the total number of employees in the company. Select count() from emp. Find the number of employees in the research department. Select count() from emp, dept. Where no number and name research". . Count the number of distinct salary values. . . Select count(distinct salary) from emp: . Retrieve the names of employees, who have a or more dependants. Select name, lnamefromempwhere(select, count() from dependant here assn son) > a. Scanned with cam scanner. For each department retrieve department number, number of employees in that department and their average salary. Select a no, count(a) , avg(salary) fromm group by do. . For each project, retrieve the project number, project name and the number of employees working on the project. Select no, name, count() from project, works on here no number group by no had name been a key, group by name would be possible. (or group by(no, name)) . For each project, on which more than a employees work, retrieve the project number, project name and number of employees who work on the project. Select no, name, count() from project, works. Nowhere number in group by number, name having count() > a usually used where count(a) has already been computed. . For each project, retrieve the project number, project name and number of employees of department of who work on the project. Select no, name, count() from project, works. On, emp here number no and group by number, name. Assn san and a no a scanned with cam scanner. . For each department having more than a employees, retrieve the department name, and the number of employees getting more than is. Of. Of a. Select name, count() from dept, emp here number do and salary> 40000 and non(select anofromempgroup by in having count() > a) . Group by name. . Reprieve a list of employees and the projects they are working on, ordered by department, and within each department alphabetically by the last name and the first name. Select name, name, name, name rodent, works. On, project, emp. Where number no anderson san and no a number. Order by name, name as, name. Ordered in the ascending order by default. Desc> descending order. . . . . Database manipulation: insert; delete; update. Scanned with capstan reinsert into emp values(richard; a, marine. . .) insert into emp(name, name, san) othe

r fields hold values(richard, marine, 2358928516') defaul
t/ full values. Delete dependant the structure remains, w
hile all constituent data are removed. Insertion hereafte
r into the table is possible. . Delete from empower name
dey. Delete from empire reason(select dnumberfromseptwher
e name research) delete all employees whose salary is les
s than the average salary. Delete from emp"marks all reco
rds to where salary< (select avg(salary) be deleted, duri
ng the from emp) course of the execution of the query; an
d the marked records the deleted physically at the end of
the query execution. It may so seem that avg(salary) is
reevaluated after each record is marked, but practically
the value is evaluated only once. An update clause is use
d to change the value of an attribute in a tuple. The whe
re clause is used to select the tuple to be updated(if th
is clause is not present> all tuples) , and a set clause
is used to no scanned with camera terminate the attribute
whose value is to be changed and to specify the a new va
lue. Update project set location mumbai, drum where numbe
r of. . Increase the salary by of of all employees in the
research department. Update else salary salary+ (a. A sa
lary) where a no in(select anumberfromdeptwhere name rese
arch) views in sol: a view is a single table that is desi
red from other tables. It does not necessarily exist in i
ts physical form. It is considered as a virtual table, in
contrast to the base tables, whose tuples are actually t
o side in the database. We can think of a view as a way o
f specifying a table that we need to reference frequently
, even though it may not exist physically. Leg, we may fr
equently use queries that retrieve the employee name and
the project names that the employee works on; rather than
having to specify the join of emp, works on and project
tables each time we issue the query, we can define a view
that is a result of these joins. We can then issue queri
es on the view, which are specified as"single table a ret
rievals", rather than are generals involving a joins on t
hree tables. A view is defined in sol, using the create v
iew command create view a as< query scanned with cam scan
ner. A: create view works on i a select name, name, name,
hours rome, works on, project anywhere in assn and same
attribute names as the tables are used in the no number.
View here, we need not specify any new attribute names. I
t inherit the names of the vico attributes from the defin

ing base tables. Emp, works on, project. Create view dept info(dept name, no of emp, tot sal) a select a name, count() , sum(salary) from emp, depth herein number group by name. . Retrieve the first name and last name of employees who work on name project of. Select name, name. My frameworks anywhere name project a a view is always update, i. A. , a we mail the tuples in the base tables, on which the view is defined, the view automatically reflects these changes. This is done by the underlined bus. If we do not need a view anymore, we can use the drop view command to dispose it off. A. A. Drop view sept info. Scanned with cam scanner updating views: considering the works. On i vireo, and suppose we issue the command to update allor ibulepname of john smith from project to project. Update works. Onset name: project where name john and name smith and name: project of: . Possible reflections of change on base: this query can be mapped into several updates in the base relations to give the desired update on the view . A) update works onset no(select number from project where name project a) where assn(select sonfromempwherefname : john and name: smith) and no(select number from project where name: project a) a) update projects name project[in cases where instances per where name: project a training to project do not exists in general, we cannot guarantee that any view can be updated, , whenever, on update on the view can be mapped to more than one, update on the under lying base, we must have a certain procedure to choose the desired update. We can make the following obs a scanned with camera innervation: a) a view with a single defining table is update able, if the view attributes contain the primary key, or some other candidate key of the base situation, as this maps each view tuple to a single base tuple. A) views defined on multiple tables using joins, are generally not up rateable. A) views defined using grouping and aggregate functions are not peaceable. As a result of these observations, most database systems impose the following constraints: a modification is permitted through views, only if the view in question is defined on terms of one relation of the actual relational database. Transaction processing: we consider an interleaved model of concurrent execution. The execution of a program that accesses or changes the content of the data base is called a transaction. In our case, it refers to a program execu tion that updates the database unless we explicitly state otherwise. Basic database operations: read(of, a) : a: program variable: database item. : a< a. Read(a) : xxx. Write(a) : a program variable database item. Fugitive operations of read(a) : primitive operations of write() . Locate data items in disk block. . Locate data item in disk block. A. Read block into main memory buffer, . Stead block into main memos if not present already. Buffer, if not present already. Locate date item in buffer and. Modify data items in buffer(from assign to program variables. Program variables) and write buffer scanned with cam scanner. Content into dick the recovery manager is a subsystem of the nos package. It decides if the buffer content is to be transferred to the dick immediately or later. (of: of mods. We have a variable(a) holding the number of bookings made on airline ais while(a) hale the same for airline as, transaction(to.) is concerned tod discarding a seats of a. . And booking those seats end as, while to is concerned with booking in seats of a. To can(a) read(a) < a xxx+ a. Consistency check: a+ write(a) write(a) constant read() . We are unaware of the or der of transactions consist write(a) ending arbitrary. Get, initial values be as follows: a of; a. A; a. A) to, and to serially executed> a a web) considering a schedule, spread(a) xxx unreal(a) a a+ max: sex a write(a) will read(a) write(a) fisc over tuition(in, last write(a) update lost) lost update problem the lost update problem occurs when two transactions, that access the same database items, have their operations interleaved in a way, scanned with capstan bertha makes the value of some database item incorrect. A) considering a schedule: , read(a) value read is dirty a since to is not yet comp a a write(a) let, > scope of change read() to a. A a+ write(a) of of. Read(a) yes value a failure(to) remains unchanged. To not complete, might need to undo all operations. Dirty read problem the dimity stead problem occurs when one transaction updates a database item and the transaction fails for some reason in future. In between, the updated item is accessed by another transaction, before it is changed to its original/ final value. A) considering a schedule that primarily aims at summary going the total transactions of the day: qum read(a) sum+ incorrect summary read(a) problems mhz bread(a) a a write(a) read(a) sum reads in corr read(a) act value of. Sum+ a:

summary(actual a) read(a) a yen write(a) scanned with cam scanner) unrepeatable read: this problem occurs when another transactions modifies a data item, read continuously by another transaction thu leading to stead two different values of the same data item, when a single value is expected. The five aforementioned problems depict the need of concurrency control. Transaction properties(acid properties) : a) atomicity a transaction is an atomic unit of processing, either performed in entity or not performed at all. A) consistency transactions must take the database from one consistent state to another. A) isolation a transaction should not make its updates visible to other transactions, until it is committed. A) durability once a transaction changes the database, and the changes are committed, these changes must never be lost because of subsequent failures. (both to and to, need to be aborted and the transactions undone in a) transaction operations transaction states. Start. Transactional remarks the beginning of transaction execution. The transaction enters the active state. Read/ write these specify read/ suite operations on the database items, that are executed as part of transactions. End. Transaction partially committed specifies read/ site operations have ended, and marks the end limit of transaction execution. At this point it is necessary to check whether the changes introduced scanned with camscannery the transaction can be permanently applied to the database, or that the transaction has to be aborted if it has violated concur renew checks or for any other reason. The concurrency control techniques require to ensure that the transaction did not interfere with other executing transactions. Also, some recovery protocol needs to ensure that a system failure will not result in an inability to record the changes in a trans action permanently. Commitcommittedthis signals successful end of the transaction, so that any changes executed by the transaction, can be safely committed to the database. If all the concurrency checks ore successful, the transaction is said to have reached its commit point and enters the committed state. About/ rollback ileitis signals that the transaction has ended unsuccessfully, so that any changes/ effects that the transaction may have applied to the database, must be undone. A transition can go to the failed state, it one of the checks fails or if it is aborted during its a five state. The transaction may then have to be rolled back. Terminated. This state corresponds to the transaction leaving the system. Transaction log: the recovery manager(system) maintains a log to keep track of a transaction operations. It permits recovery from transaction failures. The log is periodically backed up to archival storage, to quad against disk/ catastrophic failures. Typical log entries: (do not concern the operations) . [start, transaction to] > to(transaction identifier) has started exec scanned with cams connection. [write. To, a, old value, new value] see database item, presence of"old value"in the log entry allows undo transaction operations? . [read. To, a] (not a necessary log entry) [commit to] > that successfully completed and firms that its effect can be committed to the database. [abort to] > that been aborted. (sometimes, we might have to video execution of the log to acquire the present or latest database scenario due to the occurrences of failures/ catastrophe.) some recovery mechanisms perform deferred updates while others perform immediate updates. Schedules: a schedule a of on transactions: to. . . . . The, is an ord testing of the operations of the transactions, so that for each trans action to in a, the operations of to, in a, must appear in the same order in which they appear in in. However the operations from other transactions i can be interfered with the operation of to in a. [a, so, so] . . Safari(so) . > > read in tire(a) conflicting, (of) a. > write in to. A, (be) operations. A; > commit to. A(a) a, (a) a> about tip, (be) cecum(a) a(a) a, scanned with camscannerconflicting operations two operations are said to conflict if: a) they access the same database item. A) must belong to different transactions. A) a least one operation is a los ute operation. Recoverable schedule. A schedule is said to be recoverable, if no transaction tins, commits until all transactions to, that have written an item, that treads, have committed. Following of, that. , to the, we have the sequence, a, (or) to(a) however, commits prior to to, and thus so is not recoverable[a on the other hand is recoverable] . Cascading rollback; "a, (a) , a, (a) , of(a) , a. (a) , a(a) , a. (a) a, jail postponed to after commit(a) in cars ten. When an uncommitted transaction has to be rolled back, since it read an item written by a transaction that failed, the phenomenon is termed cascading rollback. As in so, to is rolled back since to, has failed and needs to be rolled back. (the isolation property ceases to hold) . A schedule is said to avoid cascading rollback, it every transaction in the schedule only reads items that were work. Cascade less transactions then by committed transactions. Schedule, read items only in committed france phone. A a a; a, (a, a) , we(a, a) , a, strict schedule: cascade len but not street another restricted schedule called struck schedule, where transactions can neither stead or write an item a, until the last transaction that wrote a has committed/ aborted. Al street are cascade less, but not vice versa. Scanned with cam scanner. Serial ability theory: the database system must control concurrent execution of transactions, to ensure that the database state remains consistent. All transactions are mutually independent as we have assumed. Say, we have transactions to, to. . . . . . The, then serially we can have a serial schedules. As we have assumed that the transactions re mutually independent, then any schedule is a correct schedule. Of, every serial schedule is correct. For schedule so; which is a non serial schedule, it is serial table as it is equivalent to a serial schedule. Definition: a schedules of a transactions is serial gable if it is equivalent to some serial schedule of the same a transactions. Clearly schedule a is not serial gable, whereas up is serial table(provided all operations work) . Equivalence of schedules: . Result equivalence: (as shown before) consider a schedules, and so with one transaction each say initially a of a; then read(a) read(a) both the transactions/ scheme a+ of; a a sales produce of a each. But write(a) write(a) this does not state their equivalence, of in this case we discard result equivalence. . Conflict equivalence? now, if we rewrite of as: sadly we have a serial schedule a: of(of) of a(of) conflicting opera i of, (a) of a(of) on scanned with capstan herself for a schedules(with the same transact. Of a(a) of a(a) of, (a) tons) all the conflicting operations are in i(a) a(a) the same order, then it is termed conflict a, (a) a eng equivalence. Of, (a) now, if a non serial schedule is conflict of, a(a) co, (a) we(a) to(a) in equivalent to a serial schedule, then, (a) this non serial schedule is always correct, cd: as a serial schedule is always correct. Here, it is evident that a is neither conflict equivalent to senor to so, as the order of t heir conflicting operations are not same. So, it is not serial table, hence not correct. Considering another schedules. : so of, (a) a a(a) here, we see that so is conflict, (a) . A. (a) equivalent to the serialized conducting operations a(a) jul(a) schedule be; thus, be is co. We(a) a, (a) greet. . Chris by(a) . Of a(a) a, (a) co a(a) cd. Definition: two schedules are said to be conflict equivalent if the order of the con flicking operations is the same in both schedules. (any two conflicting of.) . A schedules is conflict serial table if it is conflict equivalent to some serial schedule so. [a is not conflict serial table to either of so orc test of conflict serial ability : consider the following schedule with transactions: scanned with cam scanner. Construct a precedence graph as follow. A(a) : a for each transaction to; in schedule a. Create a node labeled to; in the proceed a(a) once graph. It (a) a(a) for each case in a. A(a) . Where i executes a read(a) after(a) a to site(a) executed by to, . Create(a) an edge from to, to to in the precedence see(a) graph. A(a) when to executes a write(a) after to executes a read(a) , create an edge from i to to in the precedence graph. When to executes a comte(a) after to executes a route(a) , create an edge from to to to in the precedence graph. The schedule a a serial gable if the precedence graph has no one les. Numbering of the notes is clone no follow scanned with camera merit each step we find out the terminating node in the remaining graph, i. A. , the node whose row has all is; and number/ label that node out the next highest number. We repeat this the all the nodes are successfully numbered. A a) of to to(a) cohere, after this step we do not find any other terminating node in the remaining graph; none of the rows have all a entries. Thus, we can conclude that the remaining graph has a cycle. Graph(a) has no cycles, so if it is a precedence graph of transactions, we can say that it is serial table; and its serialize reschedule is of the order i, to, to, to. Whereas, graph(a) has cycles, and if it is a schedule graph, then it is not serial table. A view reseal inability and view equivalence: two schedules, a are said to be view equivalent if the following old: a) the same set of transactions participate in a and see) for any operation a; (a) of to in a, if the value of a, read by the operation, has been written by cd; (a) of cor if it is the sci? final value o f a, before the schedule started) , the same conditions must hold for the value of a read by a, (a) of to; in a) if a(a) of to"is the last operation to cosine a in a, then a(a) of the must also be the last operation to write a in. The concept behind view equivalence is that, as long as each read scanned with camscanneroperation of a transaction steady the result of the same los ute of"ration in both schedules, the route operations of each transaction. Must produce the same results. The read operations are hence said to see the same new in both schedules. Condition a entries that the final walk operation on each data item is the same in both schedules, to the database state should be the same at the end of both schedules. A schedule a is said to be view serial table, if it is view equivalent to a serial schedule.] [a(a) , a. (a) , a. (a) , we(a) , a, . A(a] blind wail/ it, to to: views. But not conflict users a) a serial schedule represents inefficient processing, since there is no interleaving of operations from different transactions. . A) a serial table schedule gives us the benefits of concurrent execution, without giving up any correctness. A) it is practically impossible to determine, how the operation of a schedule will be interleaved beforehand(since they are typically determined by the is schedule a) . A) if transactions are executed at will, and then the result and schedule is tested for serial ability, we must cancel the effects. Of the schedules if it does not turn out to be serial gable. I clearly a/ an impractical approach] . A) a more practical approach is to determine protocols or sets of rules, that if followed by every individual transaction or it enforced by a abms concurrency control subsystem will ensure serialize ability of all the schedules, in which the transactions participate. Protocols: (ensures the isolation property of concurrent execution) a. Lock based protocols: these protocols employee the technique of locking data items top even multiple transactions from accessing the items concurrently, a lock is a variable associated with a data item lock(a) that describe scanned with cams annette status of the item with respect to possible operations that can be applied to it. Generally, there is one look for each riata item in the data base. Locks one used as a means of synchronizing the access by concur rent transactions to the database items. A) binary locking: a binary lock can have two states or values: locked(a) and unlocked(a) . A distinct lock is associated with each database items. If the value of the lock on a of, a cannot be accessed by a database operation that requests the item; if the value of the lock on a a, the item can be accessed when requested. The a following two atomic operations are wed with binary locking. Lock item(a) : a: if(lock(a) a) lock(a) a; else wait until lock(a) of and the lock manager wakes up the transaction) ; go to a; unlock item(a) : lock(a) a; if there are any waiting transactions, wake up one of them; the dams has a lock manager subsystem to keep track of and control access to locks. The binary locking scheme protocol : a a transaction to a) must issue lock item(a) before any read(a) or cos lite(a) operations are performed in to. A) must issue unlock item(a) after all read(a) or wrote(a) ope scanned with cam scanner. Skiing are completed in to. Will not issue lock item(a) if it already holds a lack an a.) will not issue unlock hem(a) unless if already heads a lock an the binary looking scheme is great active, line at most one transact: then can held a look on a given item of an instant. It more than on transaction tours to access a only for leading, we should allow them to access a simultaneously. A) multiple mode looking: in this scheme, a lock associated with an item a has three states: read locked/ shared locked(a) , write. Locked/ exclusive locked(a) and unlocked(a) the following three atomic operations are wed multiple mode locking: read lock(a) : a: if(lock(a) a) block(a) a; keep track of the number of tran no of reads(a) a; actions holding a"shared. Lock on an item. Else if(lock(a) a) no of reads(a) + + ; else wait(until lock(a) see and the lock manager wakes up the transaction) ; go to a; a write lock(a) : a: if(lock(a) a) wait (until lock(a) of and the lock unlock(a) a; manager wakes of the transact else ion) ; go to a; scanned with camscannerdalle up one of the(dating, transaction? , if any, no of jenna((a) ; if(no. Of rage(a) a) lock(a) a; wake up one of the waiting, transactions, if any; the multi mode locking scheme protocol: a a transaction to. A) must issue read lock(a) or write lock(a) before any(a) is performed in to) must issue write lock(a) before any a(a) is performed in to. A) must issue unlock(a) after all(a) and a(a) are completed in to. A) will not issue"read lock(a) and/ or lorie lock(a) in it already holds a read/ write lock on a. (may be relaxed in certain cases) . A) will not issue unlock(a) unless it already holds a read/ cos rite lock on a. An example of transactions that maintain the aforementioned protocol: scanned with camscannertitoinitially : a of; you. Rock(a) (a) to> a of a; a unlock(a) warlock(a) serial(a) schedule unlock(a) schlock(a) tax of. A for(a) tax+ yew(a) now following the schedule show unlock(a) alongside: a of; a: of. Co. Lock() of(a) as is evident, the schedule along side is not equivalent to any of thew(a) serial schedules; and is thus in unlock() correct. Observations: a) using window or multiple mode locking schemes do not guaranteeserializability of schedules. A) an additional protocol, concerning the positioning of the locking and the unlocking operations, in every transaction must be followed. A) two phase locking protocol(cpl) ; (basic) a transaction is said to follow the a pm protocol it all locking operations proceed the first unlock operation in the transaction. There are two phases in a transaction : a) an expanding/ growing phase, during which new locks on items can be acquired, but none can be released. A) a shrinking phase, during which existing locks can be release but no new locks can be acquired. Scanned with camscannerneither to, nor to follow the a pm. The cpl equivalent stands a: to, "to it can be proved that, if every lock(a) block(a) transaction in a schedule follows the(a) a() cpl protocol, the schedule is guaranteed lock(a) block(a) unlock(a) to be best liable. Unlock(a) two phase locking may limit the(a) a(a) amount of concurrency that can occur i a+ by a+ a. In a schedule. This is because, a"to(a) a (a) transaction i may not be able tore unlock(a) unlock(a a) lease an item a after it is through using it, if to must lock an additional item a later on; or conversely, to must lock the additional item a before it needs it so that it can release a. Hence a must remain locked by to until all items that the transaction needs to read or write have been locked; only can then be released by to. Meanwhile, another transaction seeking to access may be forced to wait, even thought is done with a; conversely, it it is locked earlier, than it is needed, another transaction seeking to access a is forced to wait even thought is not using a yet. (dead lock rand st ovation) (conservative/ static) : this scheme requires a transaction to lock all the items it accesses, before the transaction begins to e xecute, by or declaring its headset and write set. The headset of a transaction is the set of all items that the transaction reads, and the outset is the set of all items that it cos rites. If anyone of their declared items needed cannot be locked, the transaction does not lock any item; instead, it waits until all the items are available for locking. Is a deadlock free protocol; difficult to use in practise because of the need to or declare the headset and the a suite set. (strict) the most popular a pm scheme. It guarantees struck schedules here, a transaction it does not release any of its exclusive(co suite) scanned with camera overlooks until after it commits or aborts. Hence, no other transaction conrad or comte an item that is written by to unless to has committed, leading to a struck schedule for discover ability. Is not deadlock free, (rigorous) a more a destructive vacation of strict oil, which also qua san lees about schedules. Here, a transaction. To does not release any of its locks until after it commits or aborts, and so is easier to implement than strict oil. The conservative a pm must lock all its items before it starts, a once the transaction starts it is in its shrinking phase, whereas us porous cpl does not unlock any of its items until after it terminates(by committing a aborting) , so the transaction is in its expanding phase until it ends. Cpl bali; comparative, strict, rigorous. (dead) (a) (race the headless but lead) a deadlock situation: tit deadlock handling mechanisms block(+) a) deadlock prevention not allow(a) wing deadlocks to occur at all. Res look(a) to clive> less throughput. A(a) a) deadlock detection allow de block(a) padlocks identify reason about. Block(a) tit stock(a) wait states. Read(a) lock(a) read(a) . Deadlock prevention protocols: block(a) is used in situations concerning heavy transaction load. The various deadlock prevention schemes are: a) conservative a pm . A) assignment of an arbitrary linear ordering to each data item, and ask the transaction authors to lock items only in that order(not a practical assumption as the programmer or the system requires to be aware of the chosen order of the items. Scanned with cam scanner) use of transaction timestamps. (might be implemented by counters limited by permissible masc mum value.) here, it to(to) indicates the timestamp of the transaction to, then to(to) < to(to.) if to is an older transaction. The various scheme s under this methodology core as follows: i) wait die scheme: if to; tories to lock a but is not able since is locked by to; here, (waiting for younger) if to(to;) < to(to) then to; is allowed to waite seaport to. (to; dies) and restart it later with the same timestamp. Ii) wound wait scheme: if i tories to lock a but is not able since a is locked by to; here, (waiting for older) if to(to;) < to. A(to;) then abort to(to, wounds to) and restart it later with the same timestamp. Else, is allowed to wait. A) it can be shown that both schemes are deadlock free. A) both schemes end up aborting the younger of the two transactions, that may be involved in a deadlock. A) both techniques cause some transactions to be aborted and re started, even though those transactions may never actually cause a deadlock. A) both schemes avoid starvation, i. A. , no transaction gets aborted repetitively, since timestamps always increase and transactions are not allowed new timestamps when aborted. A transaction that is aborted, will eventually have the smallest time stamp. A) no waiting algorithm in this algorithm, if a transaction is unable to obtain lock, it is immediately rolled back and restarted after a certain scanned with cam scanner. Time delay, without checking whether a deadlock will actually occur or not. Transaction rollback and restart occurs unnecessarily. A) cautious waiting approach if to; tories to lock a but is unable as is locked by to. It, is blocked(waiting for some locked item) then abort to and restart it later. Else set status of to; as blocked and allow it to wait a it can be shown that this scheme is deadlock free, by considering the time a(to) at which each blocked transaction to was blocked. If the two transactions to; and to. Both become blocked, and to is waiting on to, then a(to,) < a(to) , since to can only wait at a time hen to is not blocked. Hence, the blocking times form a total ordering on all blocked transactions, so no cycle that causes a dead lock can occur. Of) lock time out if a transaction waits longer than a system de fined timeout, the system assumes that the transaction is dead locked and aborts it, regardless of whether a deadlock situation actually exists. In wait die, an older transaction is allowed to wait on a younger transaction, whereas, a younger transaction requesting an item held by an older transaction is aborted and restarted. Whereas, in the wound wait scheme, a younger transaction is allowed to wait on an older one, and an older transaction requesting an item held by a younger transaction preempts the younger transaction by aborting it. . Both schemes a result in aborting the younger of the two transactions, scanned with capstan bertha may be involved in a deadlock. These two techniques a me deadlock free, since in wait die transactions, transactions only wait on younger transactions so no cycle is created. Similarly, in wound. Wait, transactions only wait on older transactions son cycle is created. . Deadlock detection protocols: is used in situations concerning low transaction load. Deadlocks can be described precisely in terms of a directed graph, called a wait fer quasi. A node is created in the wait for graph area ice transaction that is currently executing in the schedule. Whenever a transaction to; is waiting to lock an item that is currently locked by a transaction to, create an edge titi. When to releases locks on the items that to was waiting for, the directed edge is dropped from the wait for graph. We have a state of deadlock if the wait for graph has a cycle. Each transaction involved in the cycle said to be deadlocked. To detect deadlocks, the system needs to maintain the wait for graph and periodically to involve an algorithm that searches for a cycle in the graph. Following the deadlock situation: (as per the transaction stated before) tits(an edge is drawn for each depend any. The graph is updated siege daily) . . Recovery from deadlock: a) victim selection: given a set of deadlock transactions, the victim selection algorithm determines which transactions to rollback to break the deadlock. It should rollback those transactions that will incur minimum cost, which depends on factors like: . How long the transaction has computed, and how much longer i scanned with capstan merrill compute to complete its designated task. . How many data items the transaction has used, and how many more the transaction needs for it to complete. . How many transactions will be involved in the roll back. . How many times a single transaction has been scaled back, (starvation prevention) . A) rollback: once the victim selection is over, it must be determined to how far this transaction should be rolled back, the simplest solution is total callback(abort) and present. However, it is more effective to scroll back the transaction only as far as necessary to break the deadlock, but,

this method requires the system to maintain additional in
formation about the state of all the sunning transactions
. A) starvation: in a system where the victim selection i
s based on cost factors, it may happen that the same tran
saction is always picked as a victim. As a result, this t
ransaction never completes its designated task. This situ
ation is called starvation. One solution to avoid starvat
ion is to include the number of roll backs in the cast fa
ctor. A) lovelock: another problem that may occur when it
uses locking is"veloce. A transaction is in a state of u
nlock, if it cannot proceed for an indefinite period of t
ime, while other transactions in the system continue norm
ally. This may accent if the waiting scheme for locked it
ems is unfair, giving porosity to some transactions over
others. The standard solution to lovelock, is to follow a
scheme that use a fife queue, where transactions are ena
bled to lock an item in the order in which they originall
y requested to lock the item. Basic timestamp ordering(to
) protocol: another approach that guarantees serial abili
ty, involves using transaction timestamps to order transa
ction executions for an equivalent serial schedule. Scann
ed with cams annette basic to protocol, associates with e
ach database item a, two time stamp values: read to() the
a read timestamp of a. This is the largest time stamp, a
mongst all transactions that have successfully read item(
a) . Write to(a) the write timestamp of a. This is the la
rgest timestamp, amongst all transactions that have succe
ssfully witter item(a) . The to protocol is as fol locus:
we have transactions to, to. To, to. To where to(a.) <
to(or) < . . . . < to(to) . If there be a schedule: to, t
o, to, to, to where: read/ route to(a) i in scenario a: t
o requests a write(a) . If(read to(a) > to(to)) > the va
lue of a that to was producing, was needed voraciously, a
nd the system assumed that, that value would never be pro
duced. Hence, the write(a) operation is rejected and to i
s aborted. . If(routers(a) > to(to)) a> that to is attem
pting to los rite an obsolete value of a. Hence, this wri
te() operation is rejected and is rolled back. Otherwise
the write() operation is executed and write to(a) a set t
o to(to) scenario a: to requests a read(a) . If(route to(
a) > to(to)) > needs to read a value of a, that was alre
ady over sultan. Hence read(a) is rejected and is rolled
back. . If(write to(a) < to(a)) > the read(a) operation

is executed and reach to(a) max read to(a) , to(to) a. Transaction to that is rolled back by the concurrency control scheme, as are salt of either a read/ write operation being issued is assigned a neil timestamp scanned with camera operand is restarted. A locking timestamp: pessimistic"one wait/ rollback. A validation protocol(love page) this protocol guarantees conflict semi liability since it ensures that, any conflicting read and write operations are executed in timestamp order. One of the problems associated with this protocol is that, it does not avoid cascading rollback. . A modification of the basic to protocol known as thomas lost rule does not enforce conflict serial ability, but it rejects fewer"a suite operations by modifying the checks for write(a) operations as follows. If(write to(a) > to(to)) then to is attempting to site an obsolete value of a. Therefore this write operation is ignored, and processing is continued without aborting to. The other two checks, al in the basic to protocol, remain identical in this case too.] write item(a) issued. It read to(a) , of(?) then about, roll act, reject operation: , if continue processing"witty. Item(a) already outdated. Any conducts resolved i write to(a) > to(to) thomas write rule makes use of view serial ability by deleting obsolete verite operations by transactions that issue them. Strict to: if read. (a) write. (a) issued, such that to(to) > write to(a) ; then a/ a delayed until to(that wrote a) has committed(i. A. , write to(a) to(to)) these protocols ensure deadlock avoidance, as no locks are issued. A. (a) , a, (a) we(a) of(a) a, cd recoverable, realizable. A (a) a(a) a. (a) (a) a. Cd recon, non realizable(incest are) a, (a) a, (a) we(a) no(a) (tel non record, sem bally> if no features thee. (a) a(a) a(a) a(a) cha> norm cow, hon serial competent else into recovery techniques: if a transaction fails after executing some of its operations, but before executing all of them, the database system must have a recovery scheme for restoration of the database to a consistent state that existed poison to the occurrence of the failure. . "to make a successful recovery from transaction failures, the system log keeps the information about the changes to data items during of transaction execution, outside the database. Of database backup and recovery from catastrophic failures: the main scanned with camscannertechnique used to handle cate atrophic failures is that of dance. Boot up, the to hole database and the log core periodically copied on a stable average media. In case of catastrophic failures, the latest backup copy can be seconded to disk, and the system can be re striped, the system log is usually substantially smaller than the database itself, and hence can be backed up more frequently. A new system log it stabile after cath backup operation. Hence, to recover from click failures: at the database is first is created on dick from its latest back up copy: by then the effects of all the committed transactions choose ope mentions have been entered in the backup copy of the system log, re then are constructed. A) other techniques: . Referred update(no undo/ reno almost item) . Immediate update(undo/ redo algorithm) [checkpoint entries in a system log checkpoints are another entry type in a system jog. The recovery manager decides at what intervals to take a checkpoint, taking checkpoints consists of the following actions: a) suspend execution of transaction temporarily. Of) force a suite all update operations of committed transactions from main memo sly buffer to disk. A) los ute a checkpoint record in the log and force write the jog to disk. Blog checkpoint enjoy: checkpoint.] ivy resume transaction execution. Recovery based on deferred update: the idea behind deferred and ate technique is to defer/ postpone any updates to the database its until the transaction completes its execution successfully and reaches its commit point the typical deferred update protocol is as follows: scanned with cam scanner. I. A transaction cannot change a database until it teaches it a com uni) , point. A. A transaction does not sirach its comment paint and all is. Update operations are a recorded in the dog and the leg is face written to dick, if a transaction fails before reaching its commit print, it wills not have changed the database in any way. Thus undo is not needed. It maybe necessary to reno the effect of the operations of a committed transaction from the log, since their effect may not have been recorded in the database i. A. , changes reflected in the care but not in the database checkpoint not taken] . Thus, such an algorithm is name a no undo/ redo algorithm. We consider a system in which concurrency control a. Is the two phase locking protocol. To combine deferred update with this protocol, we keep all the locks on items in effect until the transaction re cach es its commit point, after which all the locks are released. This ensures a strict and serial table schedule. Assume that checkpoint entries care included in the log. Recovery ming deferred update in the multiuse environment: procedure urdu elise two lists of transactions by the system committed transactions list(to) since last checkpoint. Active transactions lust(to) reno all the calcite operations of the committed transactions from the log, in the order in which they were written into the log. The transactions that are active and did not commit, are effect"ely cancelled and must be resubmitted. Redo(write of) redoing a write operation consists of examining its log entry[write to, a, new val] and setting the value of item(a) in the database to the new val. (no unto, since old val is not stored) . Scanned with cam scanner the redo operation needs to be idempotent. In fact, the whole recovery process should be idempotent, clearly, if the system tells during the recovery process the next recovery might video certain omit operations that had already been is done. The result of recovery from a crash during recovery should be the same as the result of recovery when there is no crash during recovery. To. T3t4tstimeto(checkpoint) (crash) a) to has committed before checkpoint was taken at time to, , but to and the have not committed yet. A) before the system crash, at time to, to and to have committed but to and to have not. A) thus redo of to is not necessary. A) but redo of route of of to, and to is required, as they have committed but after checkpoint time. A) to and to are ignored. They are effectively cancelled rolled back. They are to be are submitted. Instead of redoing every modification on the same data item a, it would be economical to set a to it final modified value and maintaining a redone list, such that during the bottom up traversal of the realist, is not one is done. Scanned with camscannerdisadvantages of deferred update: limits the concurrent execution of transactions, since all items re main locked until the transaction teaches its commit point. Advantages of deferred update: transaction operations never need to be undone; the seasons being: a) a transaction does not record its changes in the database until it reaches its commit print(no rollback) a) a transaction will never read the value of an item, that is position by an uncommitted transaction, since items remain locked until a transaction bl eaches is commit point. (no cascading rollback) . Recovery based on immediate update: in this case, when a transaction issues an update command, the database can be updated immediately without any need to wait for the free action to reach its commit point. An update operation must be recorded in the log on disk, before it is applied to the database, so that a me over can be easily made, the two types of immediate update are: a) if the recovery technique ensures that all updates of a transaction are recorded in the database on dick, before the transaction commits, there is never a need to redo any operation of committed(active) (committed) transactions. Such an algorithm is called undo/ no redo. Algorithm) if the transaction is allowed to commit before all ii changes(active) , (on communist) are written to the database, then we we undo/ redoaig oscitim. Considering a system in which concurrency control uses the two phase locking protocol in conjunction with the immediate update technique. Assumption checkpoints are included in the log. Recovery wing immediate update in the multiuse environment: scanned with camscannerprocedure rio elise two lists of transactions maintained by the system a list of committed transactions(to) since last check point a list of active transactions(to) undo all the a suite of of active transactions wing undo in the reverse order in which they were written into the log. Redo call polite of of committed transactions cuing redo in the order in couch they were quitter into the log. Undo(route of) undoing a cost the of consists of examining its log entry[write, a, old val, new. Val] and setting the value of a in the data love to the old val. Undoing a number of joule of from one/ more transactions from the log must proceed in the reverse order from the order in which the operations were written in the log. . Shadow paging. . A recovery technique that uses a no undo/ no redo scheme. It does not use a log, but may require a log it required by the concurrency control subsystem. Page cold) of> page of a> page of page a< of"page a page a current(working) modification. . . . . .) page a(new) (image/ shadow of page table. Of pointer on up a ion of page. A. Page table on disk) (typical pictorial representation of shadow paging) shadow paging considers the database to be made up of a number of fixed site disk pages or blocks, say a. Scanned with cam scanner page table with centaurs is constructed

, where the with page table entry points to the with data base page on disk. The page table is kept in main memory if it is not too large. When a transaction begins execution, the current page, table, whose entries point to the most recent/ current database pages on disk, is copied into a shadow page table, and his shadow page table is then saved on disk. Dusting transaction execution, the shadow page table is never modified. When a write of is performed, a new copy of the modified data base is created, but the old copy of the page is not over smitten. The cement page table entry is modified to point to the new disk block, whereas, the shadow page table is not modified and continues to point to the old disk blocks. For pages updated by the transaction, two versions are kept the old version is referenced by the shadow page table and the new version by the current page table. To recover from a failure during transaction execution, it is sufficient to free the modified database pages and to discard the current page table. The state of the database before transaction execution is is covered by gre inspecting the shade page table, so that it becomes the current page table once more. Committing a transaction corresponds to discarding the previous shadow page table and freeing the old pages on dick that it references. Clearly, this technique may be categorized as no undo/ no red technique for recovery. Advantages: there is no need to undo redo any transaction operations. Disadvantages: a) updated database pages change location on disk. Hence it is difficult to keep sedated data base pages close together on disk without complex storage management strategies. ") if the page table is large, the overhead of writing shadow page tables to disk, as transactions commit, is significant. Scanned with cam scanner) each time that a transaction commits, the database pages containing the old version of data becomes inaccessible. Such pages are considered garbage. (garbage may be created also as a side effect of crashes.) "periodically it is necessary to find all the garbage pages and to add them to the list of free pages. This process called garbage collection, imposes additional overhead and complex a city on the system. Concurrent transactions sharing pages are difficult to maintains simultaneous page tables need to be updated when such a page is up dated. Complex schemes for shadow page table maintenance required. . Referentia l integrity(nav the, of a. . Integrity constraint a(of a, of. Entity integrity. . Consistency constraint a(of) a. Bucket overflows north of a. . External hashing. Optimistic contain by contract validation protect(deadlock prevention transactions) each transaction i executer in a/ a different shares in its lifetime depending on whether it is a read only/ update fra actions. A) read phase> > i reads values of stores in local variables updates on local variables. A) validation> no termination if i to move to write phase without serialigabilityviolation; in failure here, then transaction abort. A) voila> temporary focal updates written to data our. Of(to) validation to(to) ; if to(to) < to(to) of sexual equivalent to to, the. Given is(or) < to(to) furnish(to) < start(to) > the finishes before to real or start(a;) < rush(to) < validation(to) > to completes before to start validation of write phases non overlapping a. Serial ability maintained. Read bread(a) quad against cascading rollback a by. Viral a sole only after transaction read(a) issuing while was submitted, starvation a+ a possible. : conducting from actions tempo display a+ a) < validate> write(a) write(a) or to blocked to email long trounce tonto french. Scanned with cam scanner.