1. Why would you choose a database system instead of simply storing data in operating system files? When would it make sense not to use a database system?

A database is an integrated collection of data, usually so large that it has to be stored on secondary storage devices such as disks or tapes. This data can be maintained as a collection of operating system files, or stored in a DBMS (database management system). The advantages of using a DBMS are:

- Data independence and efficient access. Database application programs are independent
  of the details of data representation and storage.
- Reduced application development time. Since the DBMS provides several important functions required by applications, such as concurrency control and crash recovery, high level query facilities, etc
- Data integrity and security. The view mechanism and the authorization facilities of a DBMS provide a powerful access control mechanism
- Data administration. By providing a common umbrella for a large collection of data that is shared by several users, a DBMS facilitates maintenance and data administration tasks. A good DBA can effectively shield end-users from the chores of fine-tuning the data representation, periodic back-ups etc.
- Concurrent access and crash recovery. A DBMS supports the notion of a transaction, which is conceptually a single user's sequential program.

2.What is logical data independence and why is it important?

Logical data independence means that users are shielded from changes in the logical structure of the data, i.e., changes in the choice of relations to be stored. For example, if a relation Students(sid, sname, gpa) is replaced by Studentnames(sid, sname) and Studentgpas(sid, gpa) for some reason, application programs that operate on the Students relation can be shielded from this change by defining a view Students(sid, sname, gpa) (as the natural join of Studentnames and Studentgpas). Thus, application programs that refer to Students need not be changed when the relation Students is replaced by the other two relations. The only change is that instead of storing Students tuples, these tuples are computed as needed by using the view definition; this is transparent to the application program.

3.Explain the following terms briefly: attribute, domain, entity, relationship, entity set, relationship set, one-to-many relationship, many-to-many relationship, participation constraint, overlap constraint, covering constraint, weak entity set, aggregation, and role indicator, relation instance, relational cardinality.

- **Attribute** - a property or description of an entity. A toy department employee entity could have attributes describing the employee's name, salary, and years of service.
- **Entity** - an object in the real world that is distinguishable from other objects such as the green dragon toy.
- **Relationship** - an association among two or more entities.

- **Entity set** - a collection of similar entities such as all of the toys in the toy department.
- **Relationship set** - a collection of similar relationships
- **One-to-many relationship** - a key constraint that indicates that one entity can be associated with many of another entity. An example of a one-to-many relationship is when an employee can work for only one department, and a department can have many employees.
- **Many-to-many relationship** - a key constraint that indicates that many of one entity can be associated with many of another entity. An example of a manyto-many relationship is employees and their hobbies: a person can have many different hobbies, and many people can have the same hobby.
- **Participation constraint** - a participation constraint determines whether relationships must involve certain entities. An example is if every department entity has a manager entity. Participation constraints can either be total or partial. A total participation constraint says that every department has a manager. A partial participation constraint says that every employee does not have to be a manager.
- **Overlap constraint** - within an ISA hierarchy, an overlap constraint determines whether or not two subclasses can contain the same entity.
- **Covering constraint** - within an ISA hierarchy, a covering constraint determines where the entities in the subclasses collectively include all entities in the superclass. For example, with an Employees entity set with subclasses HourlyEmployee and SalaryEmployee, does every Employee entity necessarily have to be within either HourlyEmployee or SalaryEmployee?
- **Weak entity set** - an entity that cannot be identified uniquely without considering some primary key attributes of another identifying owner entity. An example is including Dependent information for employees for insurance purposes.
- **Aggregation** - a feature of the entity relationship model that allows a relationship set to participate in another relationship set. This is indicated on an ER diagram by drawing a dashed box around the aggregation.
- **Role indicator** - If an entity set plays more than one role, role indicators describe the different purpose in the relationship. An example is a single Employee entity set with a relation Reports-To that relates supervisors and subordinates.
- Every operator in algebra accepts (one or two) relation instances as arguments and returns a relation instance as the result.
- Relation cardinality expresses the minimum and maximum number of entity occurrences associated; indicated by placing appropriate numbers beside entities, using format (x, y).

4. Explain the difference between logical and physical data independence.

Logical data independence means that users are shielded from changes in the logical structure of the data, while physical data independence insulates users from changes in the physical storage of the data.

Specialistion hierarchy- Entity supertypes and subtypes are organized in specialization hierarchy ; can have many sub-/supertype levels

Role of DBMS – advantages-disadvantages : 3 main functions : data definition, data manipulation, user interface (many other functions: data security, data integrity, data access sharing, data access control, data recovery)

Advantage: data independence , efficient data acces, reduced development time

Weak entity: entity is said to be existence-dependent if it can exist in database only when it is associated with another related entity occurrence; referred to as *weak* entity

Attributes:

•Should be simple and single-valued (atomic data)
•Should document default values, constraints, synonyms, and aliases
•Derived attributes should be clearly identified and include source(s)
•Should not be redundant unless this is required for transaction accuracy, performance, or maintaining a history
•Nonkeyattributes must be fully dependent on the PK attribute

One-to-many relationship:

•commonly referred to as a *parent/child* relationship
•child may have at most, one parent
•parent may have many children
•child row can be related to zero parent rows
•referred to as an *optional relationship*
•child may exist without parent

Domain -set of valid values for an attribute

Relationship Cardinality:

•One-to-zero or more
•One-to-one or more (at least one)
•One-to-zero or one (no more than one)
•One-to-some fixed range (in this case,between 4 and 8 inclusive)
•One-to-exactly N (in this case, 5)

Many-to-Many relationships:

•there would be more than one parent with more than one child
•for example, nearly any single model of car it is sold by many different car dealers; car dealer sells many different car models
•is not directly implementable using simple SQL relationship
•typically implemented by introducing another relationship table
•keys from both tables in relationship are migrated to a new table that is used to implement the relationship

Relationships

•*Child Entity* whose instances can be related to zero or one instance of the other entity (parent entity)
•*Parent Entity* whose instances can be related to a number of instances of another entity (child entity)
•*Relationship* association between two entities or between instances of the same entity
•denoted by line drawn between two entities, with solid circle at one end to indicate where the attribute is migrated to
*Identifying*primary key of one table is migrated to primary key of another -child will be dependent entity
•*Nonidentifying*primary key of one table is migrated to nonprimary key attributes of another -child will be an independent entity as long as no identifying relationships exist
•*Optional identifying*when nonidentifying relationship does not require child value
•*Recursive* when table is related to itself
•*Subtype* or *categorization*, which is a one-to-one relationship used to let one entity extend another

Relationship Degree

•indicates number of entities or participants associated with relationship
•unary relationship exists when an association is maintained within a single entity
•binary relationship exists when two entities are associated
•ternary relationship exists when three entities are associated
•higher degrees exist, they are rare and are not specifically named

Logical and Physical Data Independence:

Physical Data Independence -Application programs and ad hoc facilities are logically unaffected when physical access methods or storage structures are changed.

Logical Data Independence -Application programs and ad hoc facilities are logically unaffected when changes are made to the table structures that preserve the original table values (changing order of columns or inserting columns).

Review Questions

1. Which rel al operators could be applied to a pair of tables that are not union-compatible?
R: cross-product,selection, projection, division

2. What conditions must be met before an entity can be classified as a weak entity? Give an example

R: weak entity set=an entity that cannot be identified uniquely without considering some primary key attributes of another owner entity. An example is including department information for employees for insurance purposes.

3. What is a specialization hierarchy? Example
R: entity supertypes and subtypes are organized in a specialization hierarchy; can have many sub/supertype levels; employe-pilor,mechanic,etc; an example: pilot subtype is related to one instance of employee

4. What is an overlapping subtype? Example
R: when subtypes contain disjoint subset of the supertype entity set; example: employee can be either pilot, mechanic or accountant.

5. what is logical (and physical) independence?
R: physical independence – when application programs are unaffected when physical access methods or storage structures are changed
Logical independence – when application programs are unaffected when changes are made to the table structures that preserve the original table values (change to the table order of collums or inserting collums)

6. what is a foreign key constraint? Why is it important? What is referential integrity?
FK=a pointer to a primary key of a related table
FK important because it makes the relationship between related entityes.
Referential integrity = optional clause on creation of FK that specifies what happens to row if it has a referential relationship and is deleted from parent table or changed.

7. Role of DBMS +advantages:
R: 3 main functions : data definition, data manipulation and user interface
Many other advantages: data security, integrity, data acces sharing and control, data recovery
Advantages: data independence, efficient data access, reduced development time

8. Basic database functions that a spreadsheet cannot perform?
R: Altought spreadsheet allows creation of multiple tables, it does not support even the most basic functionality, like creating relationships or adding constrains.

10. Relationship and what types of relationship exists
R: Relationship = association between entities, Participants = participating entities
Types: involved entities, ownership, cardinality;
- 1-1, 1-M, M-N

You cannot note the expert technician constraint the FAA requires in an ER diagram. There is no notation for equivalence in an ER diagram and this is what is needed: the Expert relation must be equivalent to the Type relation.