Database Midterm

CHAPTER 1: File Based Systems | collection of application programs that perform services for the users | each program defines & manages its own data | database approach | definition of data was embedded in application programs not stored independently | no control over access and manipulation of data beyond that imposed by programs | resulted in DBMS and database |Shared collection of logically related data [(and description)designed to meet the information needs of an organization] | logically related data comprises entities attributes, and relationship of an organizations information| DDL (data definition language) permits specification of data types, structures, and any data constraints [specifications are stored in database] | DML (data manipulation language) general query language| controlled access to a database: security system, integrity system, concurrency control system, | Advantages & Disadvantages of DBMS | advantages: control of data redundancy, data consistency, more information from the same amount of data, sharing of data, improved data integrity, improved security, enforcement of standards, economy of scale, balance conflicting requirements, improved data accessibility and responsiveness, increased productivity, maintenance through data independence, increased concurrency, improved backup and recovery services| disadvantages: complexity, size, DBMS cost, additional hardware costs, cost of conversion, performance, higher impact of failure | **CHAPTER 2:** Ansi-sparc arc | External Level: users view of the database, describes the part relevant to the user | Conceptual Level: community view of database, describes data stored in database and relationships among the data | Internal Level: physical representation of the database on the pc, describes data stored in the database | Data Independence | Logical DI: immunity of external schema to changes of conceptual schema | Physical DI: immunity of conceptual schema to changes of internal schema | Conceptual Modeling: conceptual schema is the core of a system supporting all user views, complete and accurate representation of an organizations data requirements, process of constructing a detailed architecture that is independent of implementation details | System Catalog: provides description of data to enable program-data independence, metadata, “data about the data” |**CHAPTER 3:** Client-Server Arc | two tier: client & server – advantages over file server: wider access to existing databases, performance, reduction in hardware costs, reduction in communication costs, consistency | three tier: client, application server, database server – advantages over two tiers: thin client, less expensive hardware, application maintenance centralized, easier to modify or replace one tier, maps naturally to the web, easier to implement load balancing | Distributed DBMS | *logically interrelated collection of shared data & description, physically distributed over a computer network, software system that permits the management of the (*distributed database) and makes the distribution transparent to users | Cloud Computing Components| On demand self-service: consumers can obtain, configure & deploy cloud services without help from provider | Broad network access: accessible from anywhere, any standardized platform | Resource Pooling: providers computing resources are pooled to serve multiple consumers, physical and virtual resources dynamically assigned and reassigned according to consumer demand | Rapid Elasticity: providers capacity cater for customers spikes in demand and reduces risk of outages and service interruptions, can be automated to scale rapidly based on demand | Measured Service: Provider uses a metering capability to measure usage of service | Service model | Software as a Service: software and data hosted on cloud, accessed through thin client interface | Data as a Service: enables data definition in the cloud and subsequently querying, no DBMS interfaces | Database as a Service: offers full database functionality to application developers, management layer that provides continuous monitoring & configuring of the database to optimized scaling | -> how data is managed | Platform as a Service: allows creation of web applications without buying/maintaining the software or infrastructure | infrastructure as a Service: offers servers, storage, network, and operating systems, typically a virtual environment | Benefits | cost reduction, scalability, improved security, improved reliability, access to new technologies, faster development, large scale prototyping/load testing, flexible working practices, increased competitiveness | Risks | Network, System, & Cloud Provider Dependency, Lack of Control & Information

Database Midterm

CHAPTER 4: Relational Keys | Super Key: an attribute, or set of attributes, that uniquely identifies a tuple within a relation | Candidate Key: a super key such that no proper subset is a super key within the relation | Primary Key: candidate key selected to identify tuples uniquely within the relation | Alternate Keys: candidate keys no selected to be primary keys | Foreign Keys: attributes, or set of attributes, within one relation that matches the candidate key or some (possibly same) relation | Integrity Constraints | Null: represents value for an attribute that is currently unknown of not applicable for the tuple, represents the absence of a value & not the same as zero or a space | Entity: in base relation no primary key attribute can be null | Referential: if a foreign key exists in a relation, either the foreign key value must match a candidate key value of some tuple in its home relation or the foreign key must be wholly null | General Constraints: additional rules specified by users or database administrators that define or constrain some aspect of the enterprise | Views | result of one or more relational operations operating on *base relations to produce another relation | | *base relation: corresponds to an entity in the conceptual schema, tuples physically stored in database | purpose: powerful and flexible security mechanism -> hides parts of the database from certain users | updates: updates to the base relation should immediately be reflected in all views, if view is updated base relation should reflect the change, allowed if query involves a single base relation & contains a candidate key of a base relation, not allowed involving aggregation or grouping operations | **CHAPTER 5: [for all joins R : S]** Operations | unions: compatible if both tables have the same number of attributes and the corresponding attributes, in the same order, have the same datatype | set difference: defines a relation consisting of the tuples that are in R but not in S, R & S must be union compatible | cartesian product: relation that is the concatenation of every tuple of relation R with every tuple of relation S, R has n & S has m -> maximum of n*m tuples with duplicates deleted | Joins | equi: attributes being compared don't need to have the same names |natural: equijoin of two relations over all common attributes, removes all duplicate attributes | Outer: (right is opposite of left) tuples from R that do not have matching values in common with columns of S are also included in result along with all matching values | Full: add all tuples from both relations | Semi: creates relation that contains the tuples of R that participate in the join of R with S | **CHAPTER 6 & 7:** Integrity Enhancement| constraints: required data (not null), domain constraints (check), entity integrity, referential integrity, general constraints | entity: primary key of a table must contain a unique non null value for each row, one primary key clause per table | referential: foreign key is column or set of columns that link each row in child table containing foreign key to row in parent table containing matching primary key, if foreign key contains a value, that value must refer to existing row in parent table, when enforced records in the referenced table can never contain a null value for the attribute that is used as a foreign key in a referring table