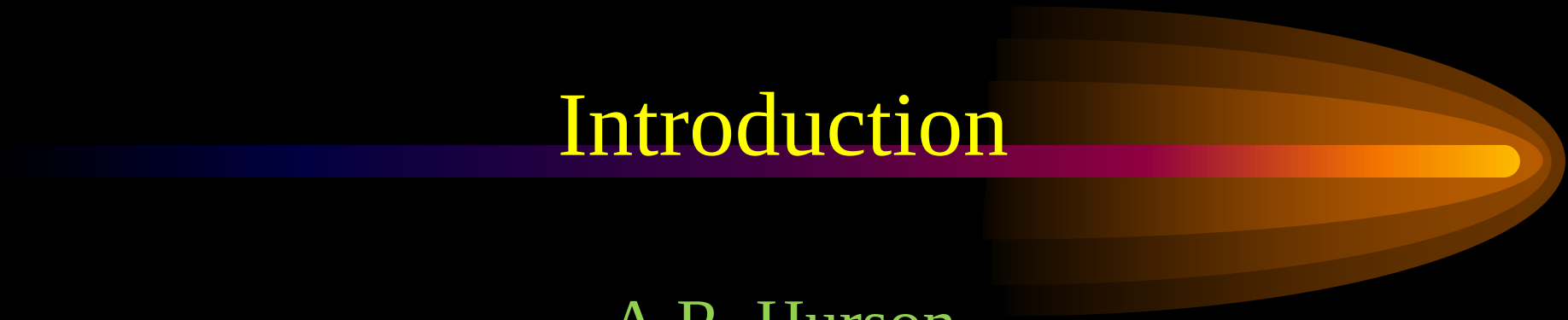
CS5300
Database Systems



Introduction

A.R. Hurson
128 EECH Building
hurson@mst.edu

Database Systems

- Instructor: Ali R. Hurson
  128 EECH Building
  hurson@mst.edu
- Text: Fundamentals of Database Systems 7th Edition (Elmasri and Navathe)
- Course material is available at:
  http://hurson.weebly.com/cs-5300-database-systems.html

Database Systems

- Grading Distribution
  - Periodic Exams and Quizzes: 40%
  - Final Exam (Comprehensive): 30%
  - Project(s) (Mandatory): 20%
  - Home works: 10%

Database Systems

- Hardcopy of homeworks and projects are collected at the beginning of class,
- It is encouraged to work as a group (at most two people per group) on homeworks and project(s) (grouping is fixed through out the semester,
- Groups are not allowed to discuss about homework assignments and project(s) with each other,
- December 1 is the deadline for filing grade corrections; no requests for grade change/update will be entertained after this deadline.

Database Systems

- Outline
  - 1. Introduction
  - 2. Relational Model
  - 3. Relational Algebra
    - a) Traditional Set Operators
    - b) Extended Relational Operators
  - 4. SQL Language
    - a) Data Definition Language
    - b) Data Manipulation Language
    - c) SQL Queries
  - 5. Performance Measure
    - a) Basic Operations
    - b) Query Optimization
  - 6. Schema Refinement and Normal Forms
  - 7. Transaction Management
    - a) Concurrency Control
    - b) Recovery
  - 8. Access Control and Data Security
  - 9. Beyond Centralized Databases
    - a) Distributed Databases
    - b) Multidatabases
    - c) Mobile Databases

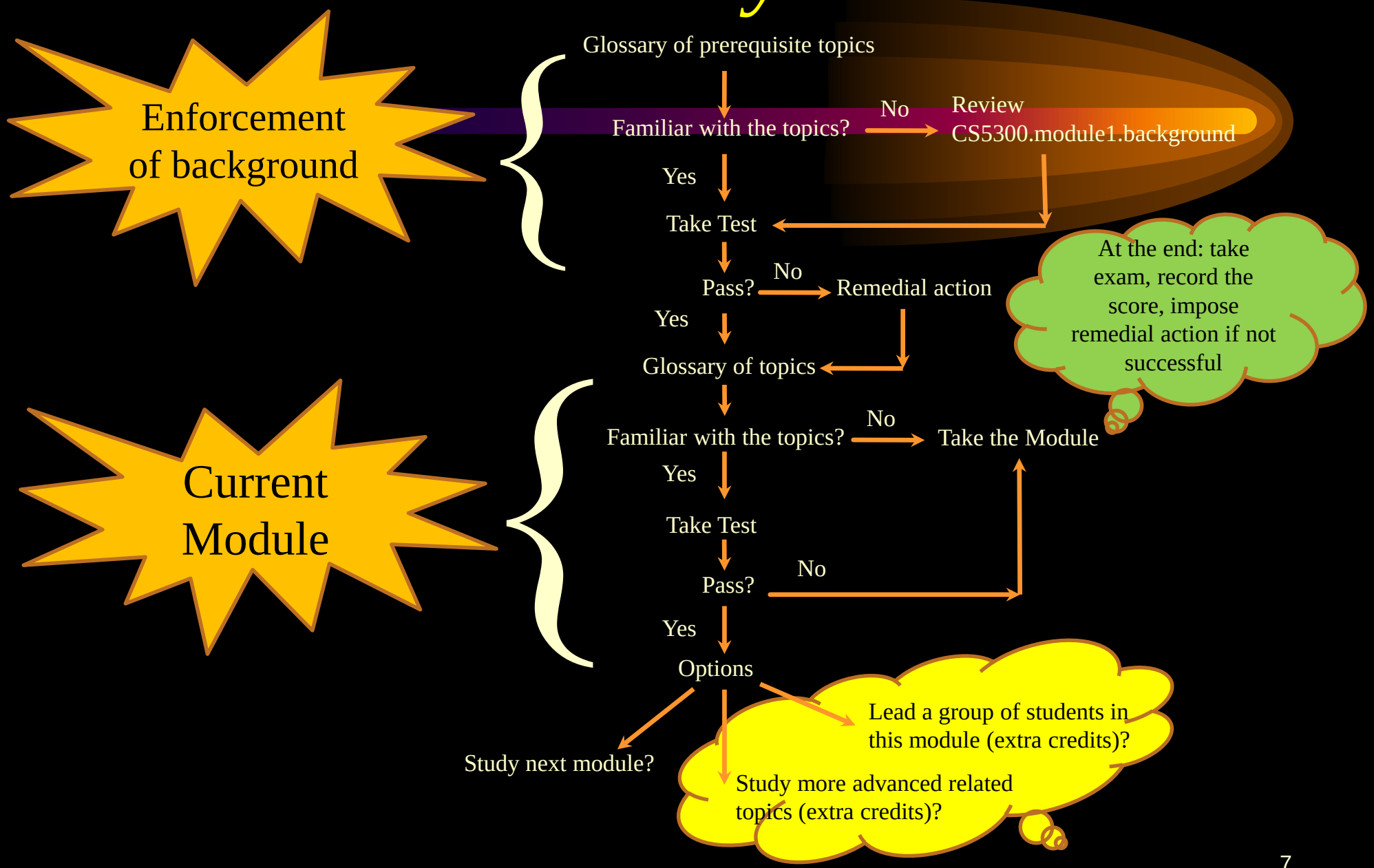
Database Systems

Note, this unit will be covered in two lectures. In case you finish it earlier, then you have the following options:

1) Take the early test and start CS5300.module2
2) Study the supplement module (supplement CS5300.module1)
3) Act as a helper to help other students in studying CS5300.module1

Note, options 2 and 3 have extra credits as noted in course outline.

Database Systems



Database Systems

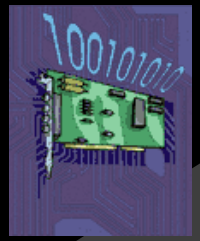
- You are expected to be familiar with:
  - Basic principles of relational database model,
  - Data Storage and Indexing
  - File organizations,
  - Some performance metrics
- If not, you need to study CS5300.module1.background

Database Systems

- This section is intended to:
  - motivate concept of databases and database management systems,
  - distinguish differences between file management system and database management system,
  - define a set of terminologies, mainly within the scope of relational model, that we will refer to extensively in this course.

Database Systems

Binary

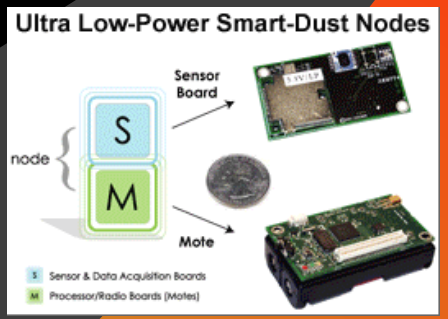


Text

Image

Multimedia

Sensors



?

Data

1970 1980 1990 2000 2010

Database Systems

Sample Data sources

In mid 1980s, it was estimated that the U.S. Patent Office and Trademark has a database of size 25 terabytes (1 tera = 10^{12}).

In 1990s, it was estimated that the NASA's Earth Observing Project will generate more than 11,000 terabytes of data.

An estimate puts the amount of new information generated in 2002 to 5 exabytes (1 exa = 10^{18}).

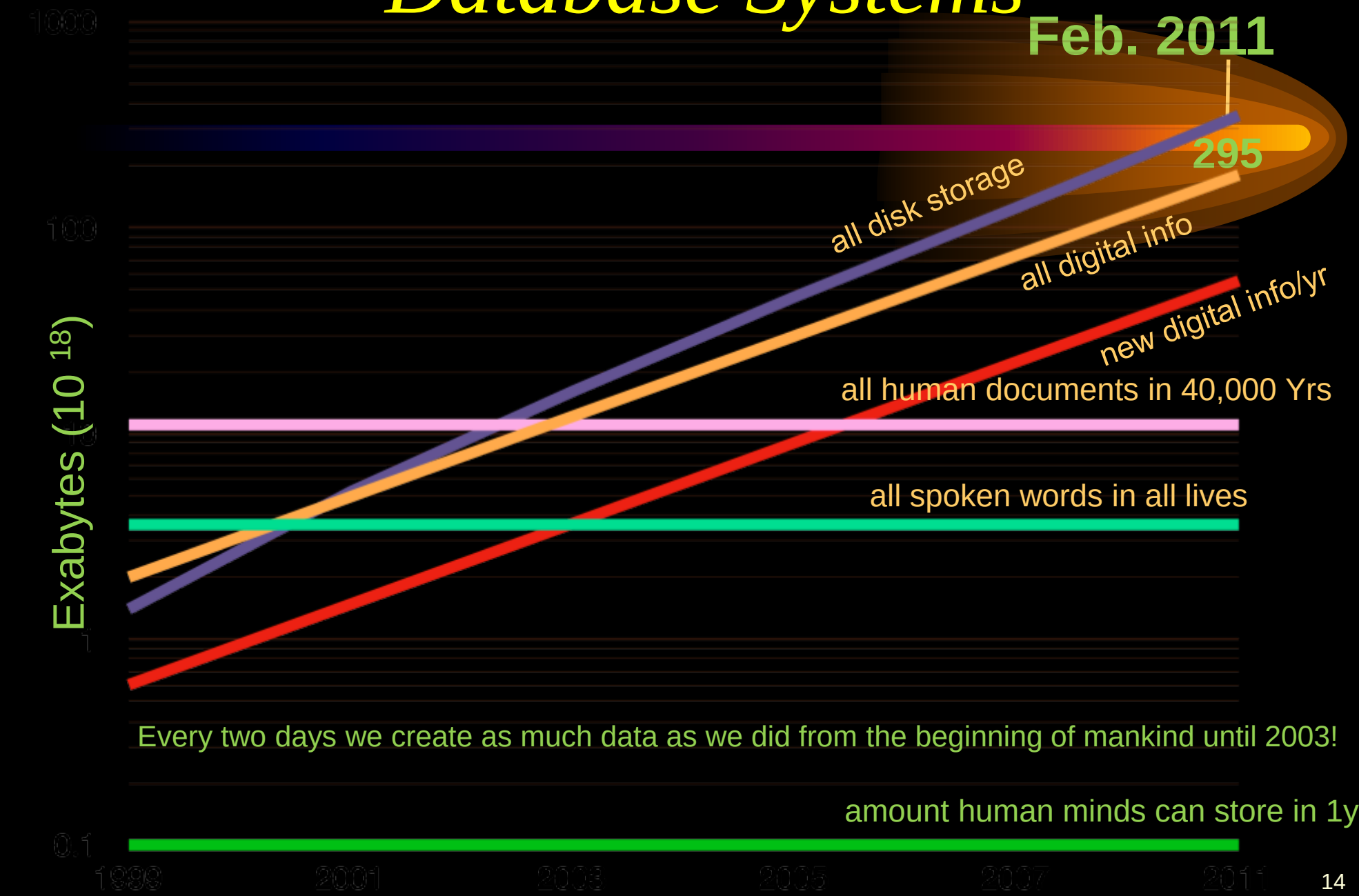
It is estimated that in 2010 more than 13 Exabyte of new data has been stored (over 50,000 times the data in the library of Congress).

Database Systems

1 Bit = Binary Digit
8 Bits = 1 Byte
1000 Bytes = 1 Kilobyte
1000 Kilobytes = 1 Megabyte (10^6)
1000 Megabytes = 1 Gigabyte (10^9)
1000 Gigabytes = 1 Terabyte (10^{12})
1000 Terabytes = 1 Petabyte (10^{15})
1000 Petabytes = 1 Exabyte (10^{18})
1000 Exabytes = 1 Zettabyte (10^{21})
1000 Zettabytes = 1 Yottabyte (102^4)
1000 Yottabytes = 1 Brontobyte (10^{27})
1000 Brontobytes = 1 Geopbyte (10^{30})

Sources: http://www.whatsabyte.com/ and http://wiki.answers.com

Database Systems



Database Systems



How many trees does it take to print out an Exabyte?

1 Exabyte could hold approximately 500,000,000,000,000 pages of standard printed text

It takes one tree to produce 94,200 pages of a book

Thus it will take 530,785,562,327 trees to store an Exabyte of data

In 2005, there were 400,246,300,201 trees on Earth

We can store .75 Exabytes of data using all the trees on the entire planet.

Database Systems

- Five V^s:
  - Volume – the sheer size of data is a major challenge and is the most easily recognizable,
  - Variety – heterogeneity of the data type, representation, and semantic interpretation,
  - Velocity – the rate at which data is generated and the time in which it must be processed,
  - Variability – the rate at which data is changing and generated (dynamic data), and
  - Veracity – Accuracy and reliability of data. Big data has to be managed in context, which may be noisy.

Database Systems

- Economical Impact
  - In 2010, it was estimated that 140,000-190,000 workers with “deep analytical” experience is needed in US alone, in addition 1.5 million managers will need to become “data-literate”.
  - It is estimated that during the period of 2012-2022, 57% of STEM jobs are in Computer Science.
  - Over the next 15 years, up to 19 million new STEM related jobs will be created worldwide and the Big Data talent pool will increase more than 500 percent by 2030. This makes Big Data related jobs the second among all STEM disciplines.

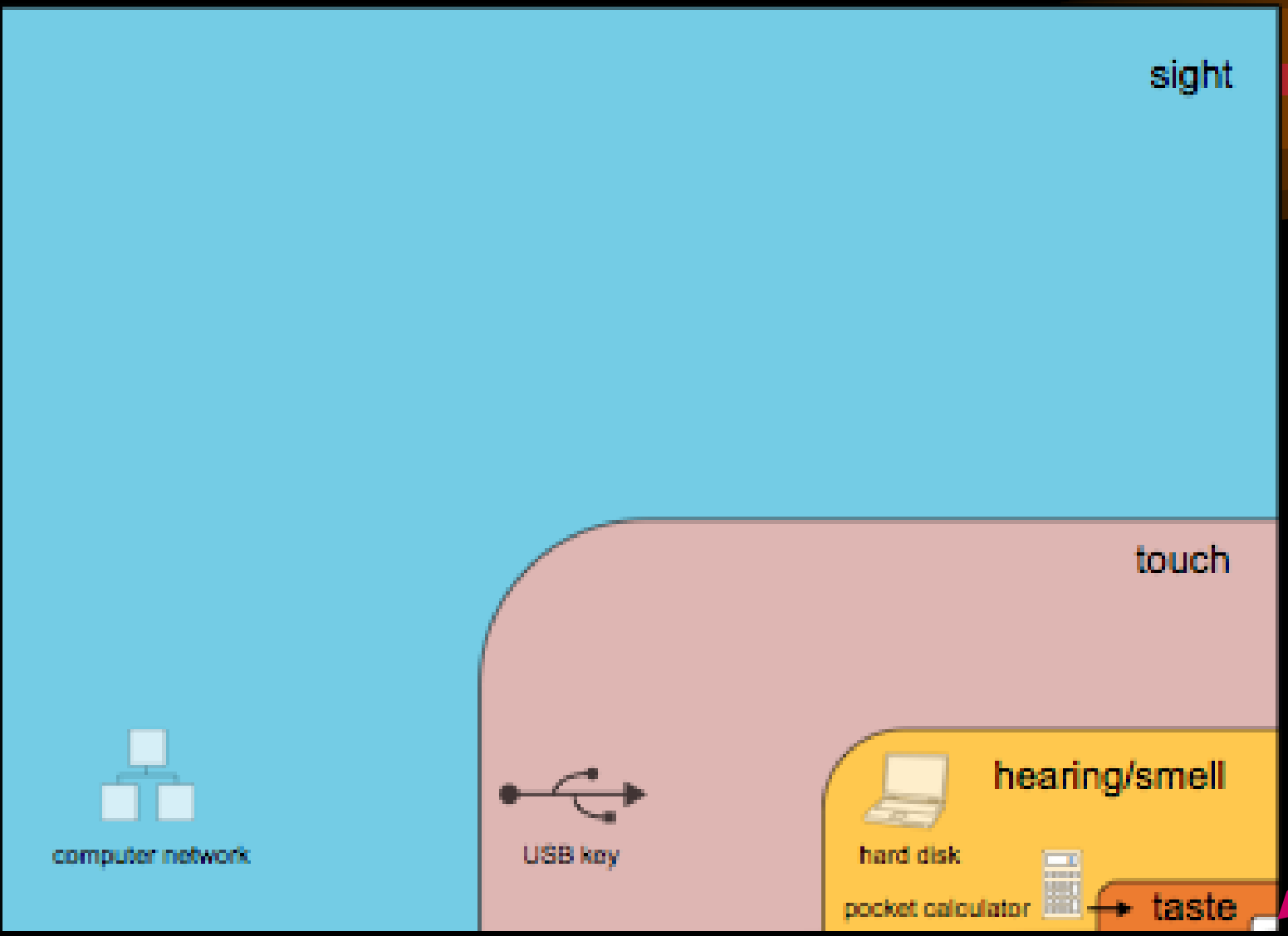
Database Systems

- Areas of application:
  - Scientific research
  - Education
  - Health care
  - Intelligent Transportation System
  - Environmental modelling
  -
  -
  -

Database Systems

- Challenges
  - Heterogeneity and Incompleteness
  - Scale
  - Timeliness
  - Privacy
  - Human Collaboration

Database Systems



Amount we actually aware of (.7%)

Source Chris Johnson, University of Utah, IPDPS2012

Database Systems

- Database systems are the outgrowth of the traditional file management systems. It is essentially a computerized record-keeping system — it is a repository for a collection of computerized data files.

Database Systems

- The user will be given facilities to perform a variety of operations such as:
  - Adding new files to the database,
  - Inserting new data into existing files,
  - Retrieving data,
  - Updating data,
  - Deleting data,
  - Removing existing files from the database.

Database Systems

- In short, database systems are designed to provide timely and reliable access to information.
  - Timely could imply; efficiency, speed, performance, robustness, ….
  - Reliable could imply; performance, security, integrity, accuracy, ….

Database Systems

- Database is an integrated, time dependent, shared collection of data describing the activities of one or more related organizations.
  - Integrated means that the database can be thought of as a unification of several otherwise distinct data files.
  - Time dependent means that the contents of database changes over time.
  - Shared implies a multi-user environment.

Database Systems

- A database system is defined by:
  - A set of data to represent entities (objects) and interrelationships among them,
  - A set of rules to preserve the data integrity, and
  - A set of operations to manipulate the data.

Database Systems

- Database management system (DBMS) is a collection of system routines designed to maintain and utilize databases — an umbrella shielding database user from hardware/software details.

Database Systems

- In a file management system data is a collection of information sources. It was designed to support efficient storage and access to the data. In short, it mainly addresses the organization of data on secondary storage.

Database Systems

- A file management system:
  - Does not provide data independence.
  - Is not a robust environment.
  - Could be very efficient for a specific application.
  - In general, is not efficient for a group of applications.

Database Systems

- In contrast to a file management system, a database management system takes a top down approach to process the information. It provides:
  - Data independence, insulates application programs from data representation and storage details (physical data independence). Furthermore, changes in logical structure of data is hidden from applications (logical data independence).

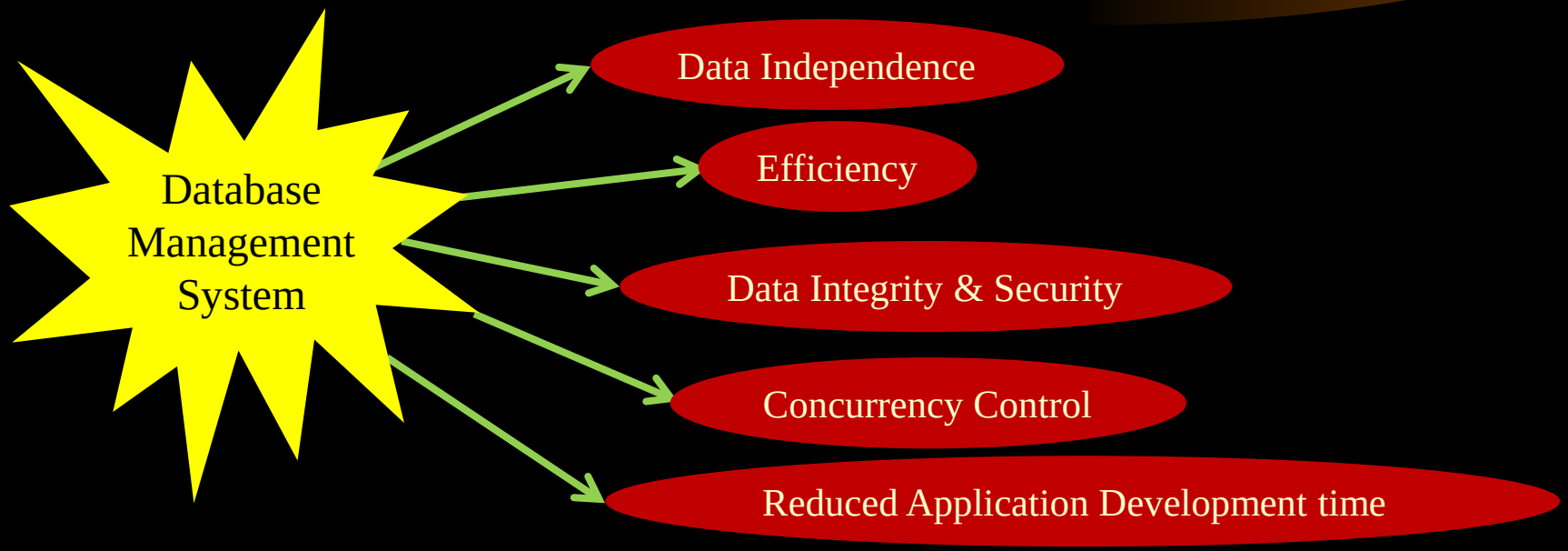
Database Systems

- Efficiency (storage and execution time), this feature is traced back to the file systems;
  - Redundancy can be reduced,
  - Data can be shared,
  - Conflicting requirements can be balanced.
- Data integrity and security (integrity constraints)
  - Inconsistencies can be avoided,
  - Security restrictions can be applied,
  - Integrity can be maintained.

Database Systems

- Data administration
- Concurrent accesses and recovery
- Reduced application development time.

Database Systems



Database Systems

- However, database management systems:
  - May not be efficient for specific application domains,
  - Are robust within the set of applications they are intended for.

Database Systems

- Questions
  - Why transition from centralized databases to distributed databases?
  - Why break a file into several smaller files?
  - Redundancy, is it good or bad?

Database Systems

- A Database management system:
  - Stores proper information on secondary storage.
  - Defines methods to identify relevant data for an application.
  - Allows to write application programs to carry out tasks for each application (user).
  - Develops schemes to guarantee data integrity (consistency).
  - Provides means for data security, privacy, and protection.
  - Makes sure that after proper modification, new state of database is preserved on permanent storage medium.

Database Systems

- Data model — a method of describing data:
  - Relational data model — Database is represented by a set of tables (relations), in which a row (tuple) represents an entity (object, record) and a column corresponds to an attribute of an entity.
  - Schema — A description of a particular collection of data using a given data model.

Database Systems

- Levels of abstraction
  - Conceptual (logical) level describes stored data in terms of the data model's conceptual data base design. This provides physical data independence.
  - Physical level specifies additional storage details. In another words, it is the realization of conceptual schema level.
  - External level (user's view of data), this provides logical data independence.

Database Systems

- Transaction is a sequence of operations that transfers database from one consistent state to another consistent state. A transaction can be looked at as a collection of read and write operations terminated by a commit or abort operation.

Database Systems

- To provide a timely access to information, a database management system should interleave several transactions and allow concurrent execution of transactions. This also means that proper concurrency rules (constraints) must be applied so that each transaction appears to execute in isolation.

Database Systems

- In general, a database management system should guarantee ACID property of a transaction (Atomicity, Consistency, Isolation, Durability):

Database Systems

- **Atomicity:** Either all operations of a transaction happen or none happen. State changes in a transaction are atomic.
- **Consistency:** A transaction produces results consistent with integrity requirements of the database.
- **Isolation:** In spite of the concurrent execution of transactions, each transaction believes it is executing in isolation. Intermediate results of a transaction should be hidden from other concurrently executing transactions.
- **Durability:** On successful completion of a transaction, the effects of the transaction should survive failures.

Database Systems

- As noted before, database systems are designed to provide timely and reliable access to information. Within the scope of transaction processing:
  - Timely means interleaving the actions of several transactions and concurrent execution of transactions (Consistency, Isolation).
  - Reliable means Atomicity and Durability.

Database Systems

- A relational database is a collection of relations.
- A relation corresponds to an entity which is referred to as a table (a collection of rows of the same structure. It is a two dimensional array (m_*n) of *m* distinct rows and *n* columns. *m* is called the cardinality and *n* is called the degree (arity).

Database Systems

- A tuple corresponds to each row of the table, it is a collection of *n* attributes.
- The primary key is an identifier for the table, a column or a collection of columns, that uniquely identifies each tuple.

Database Systems

- Domain is a pool of values from which one or more attributes draw their actual values. In another words, a domain is a collection of smallest semantic unit of data (Scalar data).
  - A domain is a set of scalar values of the same type. Note at this point we assume that a domain does not contain a null element (this will be relaxed later).
  - A domain can be classifies as:
    - Simple domain
    - Complex domain

Database Systems

- Simple domain: Domains of scalar values,
- Complex domain: A collection of simple domains.
- Note: if two attributes draw their values from the same domain, then it make sense to perform various operations (such as comparison, join, …) involving these two attributes.

Database Systems

- Relation: A relation *R* on a set of domains D_1, D_2, …, D_n (not necessarily all distinct) is a subset of Cartesian products of D_1, D_2, …, D_n.

  $R \subseteq D_1 * D_2 * .... * D_n$

  In another words, R is a collection of tuples each of *n* elements (a_i $1 \leq i \leq n$) where

  $a_i \in D_i$ for $1 \leq i \leq n$.

Database Systems

- A relation consists of two parts:
  - Heading (Schema, relation schema) — Column heads: consists of a fixed set of attributes, more precisely, attribute domain pairs;

    $\{(A_1:D_1), (A_2:D_2), \ldots, (A_n:D_n)\}$, where A_is are distinct. A schema represents the relation name, name of each attributes and the domain of each attribute.

    Students (Sid:string, name:string, login:string, age:integer, gpa:real)

Database Systems

- Body (relation instance): consists of a time-varying set of tuples. Each tuple consists of a set of attribute value pairs.

$$\{(A_1:vi_1), (A_2:vi_2), \ldots, (A_n:vi_n)\}$$

- Note cardinality of a relation may change in time, but its degree (arity) does not.

Database Systems

- For example:

S#	Sname	Status	City
S_1	Smith	20	London
S_2	Jones	10	Paris
S_3	Blake	30	Paris
S_4	Clark	20	London
S_5	Adams	30	Athens

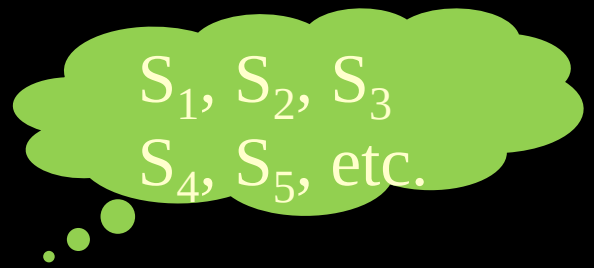
Database Systems

- The previous example shows a relation with cardinality of 5 and degree of 4. Each row represents a tuple and each column represents an attribute. Furthermore, each column draw its value from its domain.

Database Systems

- Refer to our previous example:
  - City is an attribute that draws its values from a domain which contains

  - Similarly, S# draws its value from a domain which contains



Database Systems

- Properties of relations:
  - There is no duplicated tuples. No two rows are identical — there is always a primary key.
  - Tuples are unordered.
  - Attributes are unordered.
  - All attribute values are atomic (we might relax this later).

Database Systems

- Formally, let R $(f_1{:}D_1)$, $(f_2{:}D_2)$, …, $(f_n{:}D_n)$ be a relation schema and for each f_i $(1 \leq i \leq n)$ let Dom_i be the set of values associated with the domain D_i. An instance of *R* that satisfies the domain constraints in the schema is a set of tuples (a relation) with *n* fields:

  $\{<f_1{:}d_1>,<f_2{:}d_2>, \ldots,<f_n{:}d_n> | d_1 \in Dom_1, d_2 \in Dom_2, \ldots, d_n \in Dom_n \}$

- <Sid:5,000>,<name:Dave>,<login:dave@cse.psu.edu>, <age:23>,<gpa:3.45> is a tuple.

Database Systems

- Kinds of relations:
  - Base relation is a relation that is sufficiently important (carry important information) that the database designer has decided to made it a part of the database.
  - View is a derived relation, it does not have any separate, distinguishable stored data of its own (logical relation).
  - Snapshot like a view is a derived relation. However, unlike view, a snapshot is real and not virtual — it is defined by its definition and its own stored data.

Database Systems

- Query result is an output relation resulting from some specified query.
- Intermediate result is a relation that is the result of a relational expression nested within larger expression.
- Temporary relation is a base, view, or snapshot created during the course of operations and destroyed automatically at some appropriate time by the system.

Database Systems

- Creating and Modifying Relations Using SQL:
  - CREATE TABLE is a statement that allows one to create a new table (relation);

```
CREATE TABLE Student ( Sid     CHAR(20),
                       name    CHAR(30),
                       login   CHAR(20),
                       age     INTEGER,
                       gpa     REAL )
```

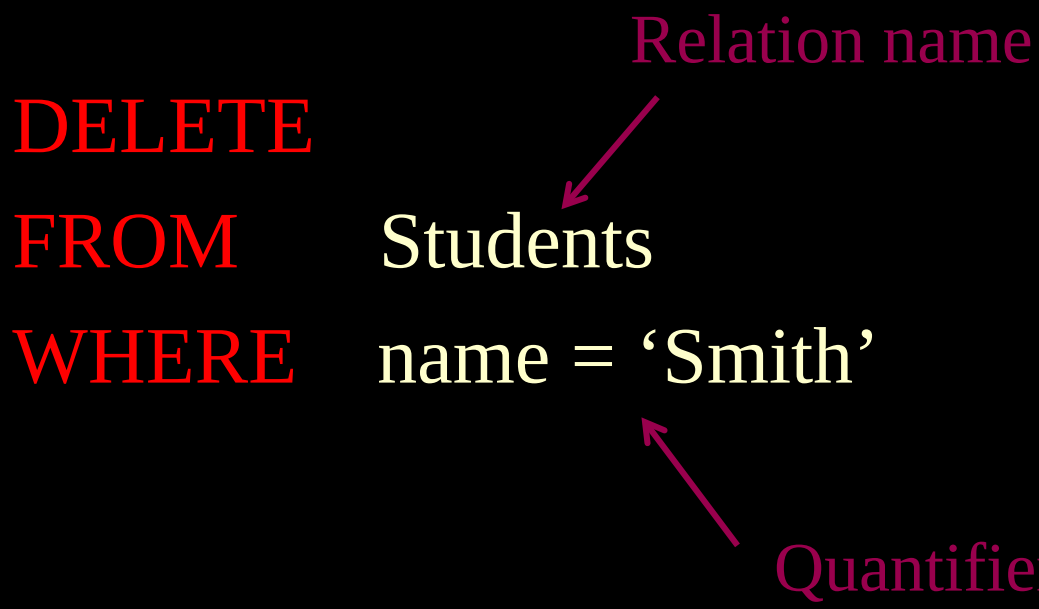
Database Systems

- INSERT command is used to insert a single tuple into a created relation (table):


```
INSERT
INTO Student (Sid, name, login, age, gpa)
VALUES      (53688, ‘Smith’, ‘smith@cs,
                18, 3.2)
```

Database Systems

- DELETE command eliminates a tuple from a relation:



Database Systems

- UPDATE command allows one to modify the attribute value of a tuple:

Relation name

```
UPDATE Student
SET    age = age + 1, gpa = gpa + .1
WHERE  Sid = 53680
```

Quantifier

Database Systems

- Integrity Rules
  - The database definition needs to be extended to include certain constraints (integrity rules — reliable operations). In the real world, for example, weight cannot be negative. Integrity rules are needed so that it can prevent such impossible configuration of values from occurring.

Database Systems

- Integrity Rules
  - Most databases will be subject to a very large number of such integrity rules.
  - Note that most of the integrity rules are specific in the sense that they are applied to one specific database.

Database Systems

- Integrity Rules
  - In this section, we will discuss about three "general integrity rules" that applies to every databases.
  - Note, the integrity rules both general and specific are defined for "base relations".

Database Systems

- Integrity Rules
  - Consider the following three relations.
  - Note that Primary key is allowed to be composite. It is possible to have all the attributes of the relation to represent the primary key — All key relation.
  - It is also possible for a relation to have more than one unique identifiers — Multiple candidate keys. In this case, one of the candidate keys will be chosen as the primary key and the rest will be chosen as alternative keys.

Database Systems

- Integrity Rules

S#	Sname	Status	City
S_1	Smith	20	London
S_2	Jones	10	Paris
S_3	Blake	30	Paris
S_4	Clark	20	London
S_5	Adams	30	Athens

Database Systems

- Integrity Rules

P#	Pname	Color	Weight	City
P_1	Nut	Red	12	London
P_2	Bolt	Green	17	Paris
P_3	Screw	Blue	17	Rome
P_4	Screw	Red	14	London
P_5	Cam	Blue	12	Paris
P_6	Cog	Red	19	London

Database Systems

- Integrity Rules

S#	P#	QTY
S_1	P_1	300
S_1	P_2	200
S_1	P_3	400
S_1	P_4	200
S_1	P_5	100
S_1	P_6	100
S_2	P_1	300
S_2	P_2	400
S_3	P_2	200
S_4	P_2	200
S_4	P_4	300
S_4	P_5	400

Database Systems

- Integrity Rules
  - Candidate Key: Attribute *K* (possibly composite) of relation *R* is a candidate key for *R*, if and only if it satisfies the following two time-independent properties:
    - Uniqueness: At any point in time, no two tuples of *R* have the same value for *K*.
    - Minimality: If *K* is composite, then no components of *K* can be eliminated without destroying the uniqueness property.

Database Systems

- Integrity Rules
  - No component of the primary key of a base relation is allowed to be null (undefined). This may not be applied to alternative keys.
  - Primary key allows associative addressing.

Database Systems

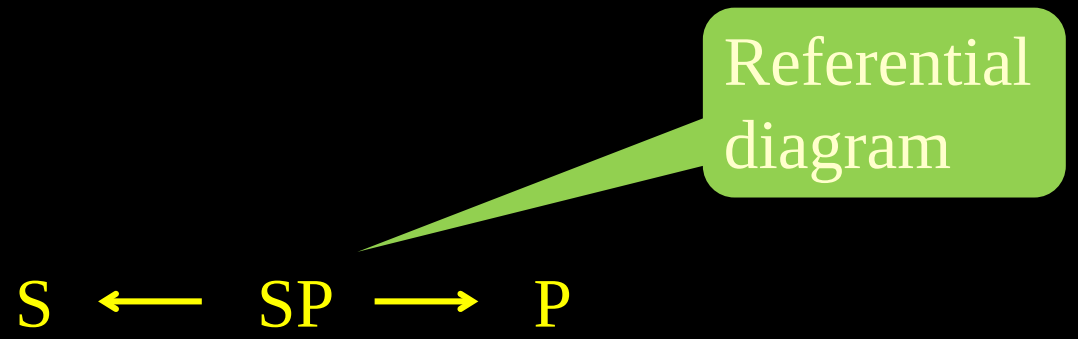
- Integrity Rules
  - Foreign Key is an attribute of a relation R_2 (possibly composite) whose values are required to match those of the primary key of another relation R_1.
  - A foreign key is a mechanism to establish references between two relations. This brings an issue know as the referential integrity.
  - Attributes S# and P#, for SP relation, are examples of foreign keys.

Database Systems

- Integrity Rules
  - Attribute *FK* (possibly composite) of base relation R_2 is a foreign key, if and only if it satisfies the following time-independent properties:
    - Each value of *FK* is either wholly null or wholly non-null — (all null or none null).
    - There exists a base relation R_1 (target relation) with primary key *PK* such that each non-null value of *FK* is identical to the value of *PK* in some tuple of R_1.

Database Systems

- Integrity Rules
  - A foreign key value represents a reference to the tuple containing the matching primary key value. The relation that contains the foreign key is the referencing relation and the relation that contains the corresponding primary key is the target (referenced) relation. In our example:



Database Systems

- Integrity Rules
  - In the following example:

    DEPT (DEPT#, …, BUDGET, …)

    EMP (EMP#, …, DEPT#, …, SALARY, …)

    DEPT# is the foreign key in EMP relation and it is not a part of the primary key of EMP relation.

    DEPT ⟵ EMP
  - A relation can be both a referenced relation and referencing relation.

    $R_3 \longrightarrow R_2 \longrightarrow R_1$

Database Systems

- Integrity Rules
  - Referential Integrity rule: The database must not contain any unmatched foreign key value.
  - Foreign key rules: At each moment in time, the database should be in a valid state and any transaction should transfer the database from a valid state to another valid state. Therefore, for an operation that violates these constraints, either the operation should be denied, or the database automatically, should generate a compensating operation.

Database Systems

- Questions
  - Can a foreign key accept nulls?
  - What should happen on an attempt to delete the target of a foreign key reference?
  - What should happen on an attempt to update the primary key of the target of a foreign key reference?

Specifying Key Constraints in SQL

- Primary Key
  - The key words UNIQUE, CONSTRAINT, and PRIMARY are used to define candidate keys. At most, one of these candidate keys can be declared as Primary Key.

Database Systems

- Primary Key

```
CREATE TABLE Students ( Sid      CHAR(20),
                        name     CHAR(30),
                        login    CHAR(20),
                        age      INTEGER,
                        gpa      REAL ,
                        UNIQUE (name, age),
                        CONSTRAINT Studentskey
                        PRIMARY KEY (Sid) )
```

Studentskey is called the constraint name – It will be returned if the constraint is violated.

Database Systems

- Foreign Key
  - Assume that in addition to the *Students* relation, we have:

    Enroll (Sid: string, Cid: string, grade: string)
  - Here Sid of *Enroll* relation should also appear in the Sid field of some tuples in the *Students* relation. The Sid field of *Enroll* relation is called a foreign key.

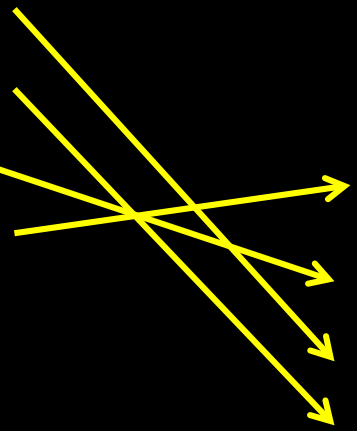
Database Systems

- Foreign Key
  - The foreign key of the referencing relation (Enroll) must match the primary key of the referenced relation (Students), must have the same number of columns and compatible type, though the column names can be different.

Database Systems

Cid	grade	Sid
Carnatic101	C	53831
Reggae203	B	53832
Topology112	A	53650
History105	B	53666

Referencing relation



Sid	name	login	age	gpa
50000	Dave	dave@cs	19	3.3
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53831	Maday	maday@music	11	1.8
53832	Guldu	guldu@music	12	2.0

Referenced relation

Database Systems

- Foreign Key
  - Key words FOREIGN KEY and REFERENCE are used to specify this constraint:

```
CREATE TABLE Enroll ( Sid       CHAR(20),
                      Cid       CHAR(20),
                      grade     CHAR(10),
                      PRIMARY KEY (Sid, Cid),
                      FOREIGN KEY (Sid)
                      REFERENCES Students)
```

Referenced relation

Database Systems

- General Constraints
  - To preserve data integrity, we may need to define domain specific integrity rules. These rules can be enforced in the form of table constraints or assertions.
    - Table constraints are defined for each relation and are enforced whenever contents of the relation is modified (insert, delete, update).
    - Assertions involve several relations and are enforced whenever any of these relations is modified.