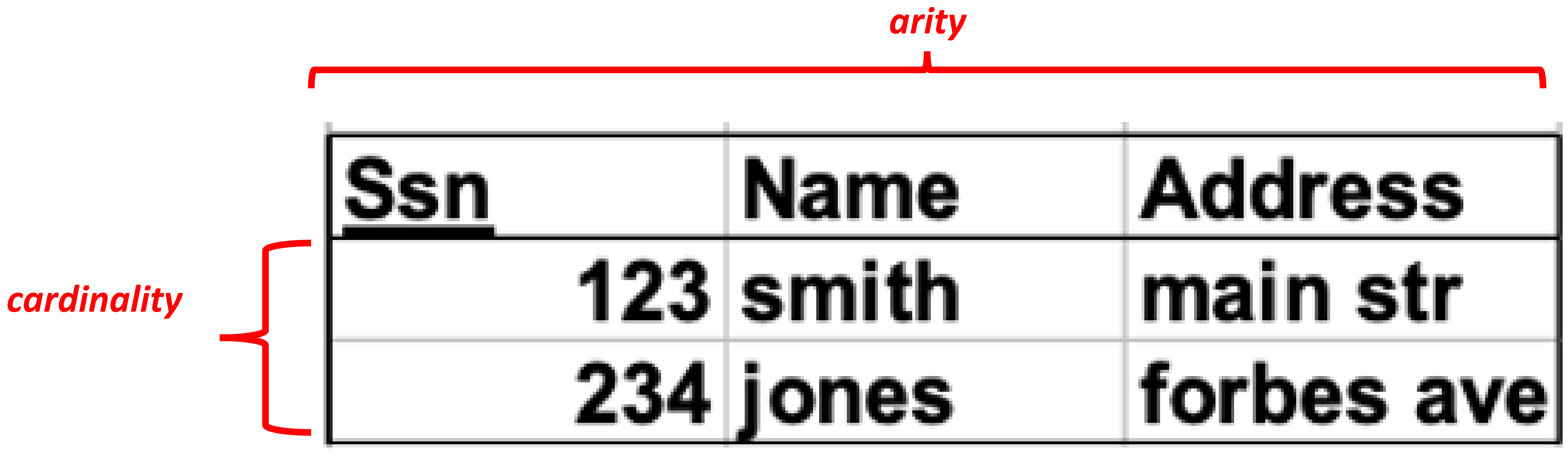
CSE 412 Database Management

Dr. Bharatesh Chakravarthi (BC)
Assistant Teaching Professor
School of Computing and Augmented Intelligence
Arizona State University

The Relational Model

What is a Relation?

- *Relation: made up of 2 parts:*
  - *Schema: specifies name of relation, plus name and type of each column.*
  - *Instance: a table with rows and columns.*
    - *#rows = cardinality*
    - *#columns = arity*


arity
cardinality

Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

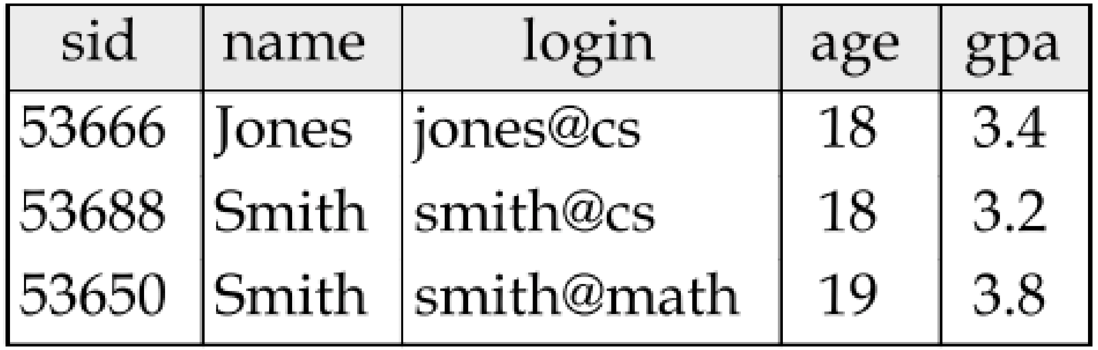
Take-aways..

- *A database is a collection of* ***relations*** *(or* ***tables****)*
- *Each relation has a set of* ***attributes*** *(or* ***columns****)*
- *Each attribute has a name and a* ***domain*** *(or* ***type****)*
- *Each relation contains a set of* ***tuples*** *(or* ***rows****)*

What is a key?

- *A set of attributes **K** is a key for a relation R if*
  - *In every instance of R, no two distinct tuples have the same values for all attributes in K*
    - *That is, **K** can serve as a "tuple identifier"*
    - *If K can serve as a "tuple identifier", it is called as a Superkey*
  - *No proper subset of K satisfies the above condition*
    - *That is, **K** is minimal*
    - *Minimal superkey is called as a key*
- *Example:*

  Student (sid, name, login, age, gpa)
  {sid, name}: superkey
  {sid}: superkey, AND key
  {name}: not superkey



sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@cs	18	3.2
53650	Smith	smith@math	19	3.8

More examples..

uid	name	age	pop
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3
…	…	…	…

- *Is name a key of User?*
  - *Yes? Seems reasonable for this instance*
  - *No! User names are not unique in general*

More examples..

uid	name	age	pop
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3
…	…	…	…

- *Is name a key of User?*
    - *Yes? Seems reasonable for this instance*
    - *No! User names are not unique in general*
- *{uid} is the key*

More examples..

gid	*name*
abc	Book Club
gov	Student Government
dps	Dead Putting Society
…	…

- *Group (gid, name)*
- *Which set of attributes is the key?*

More examples..

gid	*name*
abc	Book Club
gov	Student Government
dps	Dead Putting Society
…	…

- *Group (gid, name)*
- *{gid} is the key*

More examples..

uid	*gid*
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

- *Member (uid, gid)*
- *Which set of attributes is the key?*

More examples..

uid	gid
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

- *Member (uid, gid)*
- *{uid, gid} is the key*
- *A key can contain multiple attributes*

Primary Keys..

- *A relation can have multiple keys!*
  - *Student: {ssn}, {student-id}, {driving-license#, state}*
  - *Employee: {ssn}, {employee-id}, {phone#}*
  - *computer: {mac-address}, {serial#}*
- *DBA (Database Admin) typically picks one as the "primary" key, and underline all its attributes*
- *Other keys are called candidate keys*

Primary Keys..

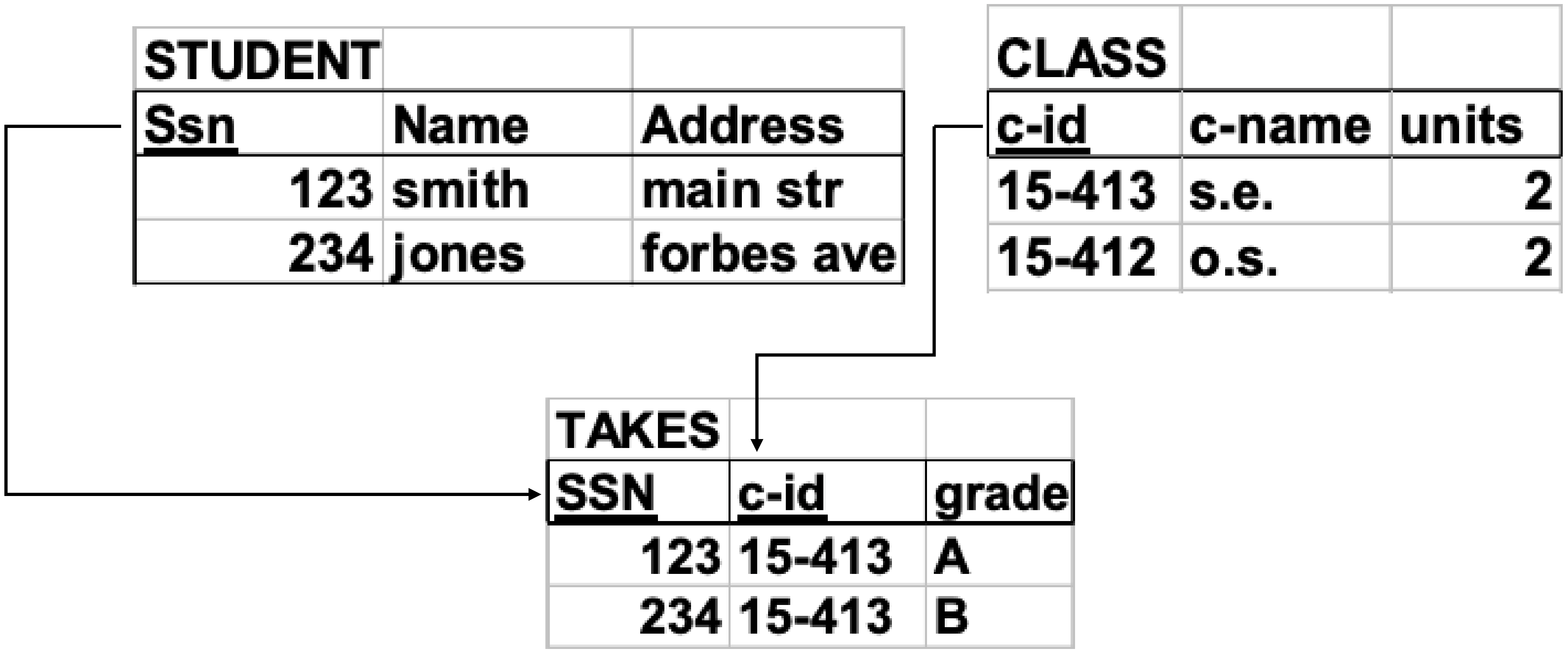
- *Example:*
  - ***Address (street_address, city, state, zip)***
    - ***{street_address, city, state}***
    - ***{street_address, zip}***
  - *DBA typically picks one as the "primary" key, and underline all its attributes, e.g.,* ***Address (street_address, city, state, zip)***
  - *Other keys are called candidate keys.*

    e.g., ***{street_address, city, state}***

What is a Database?

*A database is a collection of **relations** (or **tables**)*

Example 1



STUDENT		
Ssn	Name	Address
123	smith	main str
234	jones	forbes ave

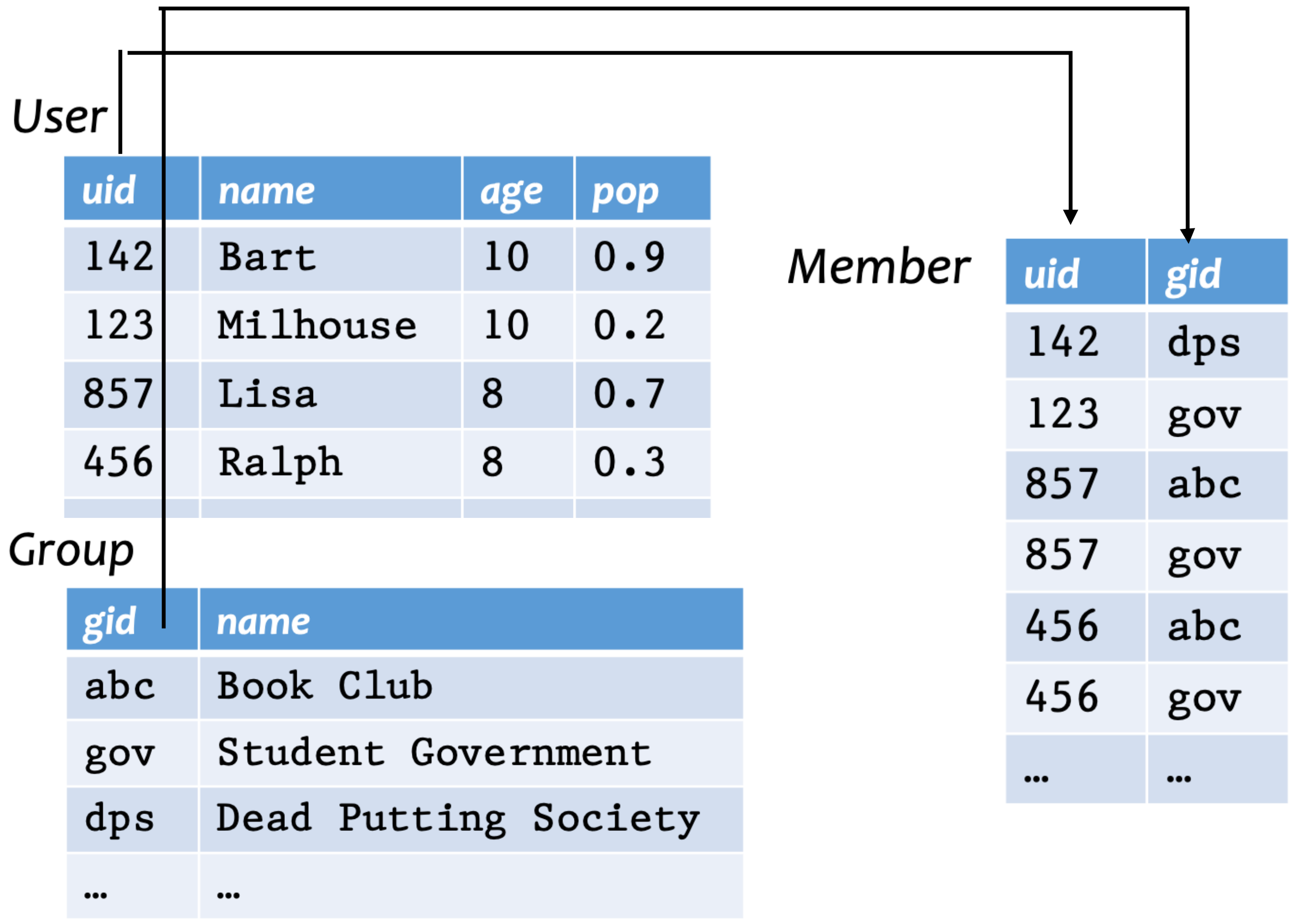
CLASS		
c-id	c-name	units
15-413	s.e.	2
15-412	o.s.	2

TAKES		
SSN	c-id	grade
123	15-413	A
234	15-413	B

What is a Database?

*A database is a collection of **relations** (or **tables**)*

Example 2



User

uid	*name*	*age*	*pop*
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3

Group

gid	*name*
abc	Book Club
gov	Student Government
dps	Dead Putting Society
…	…

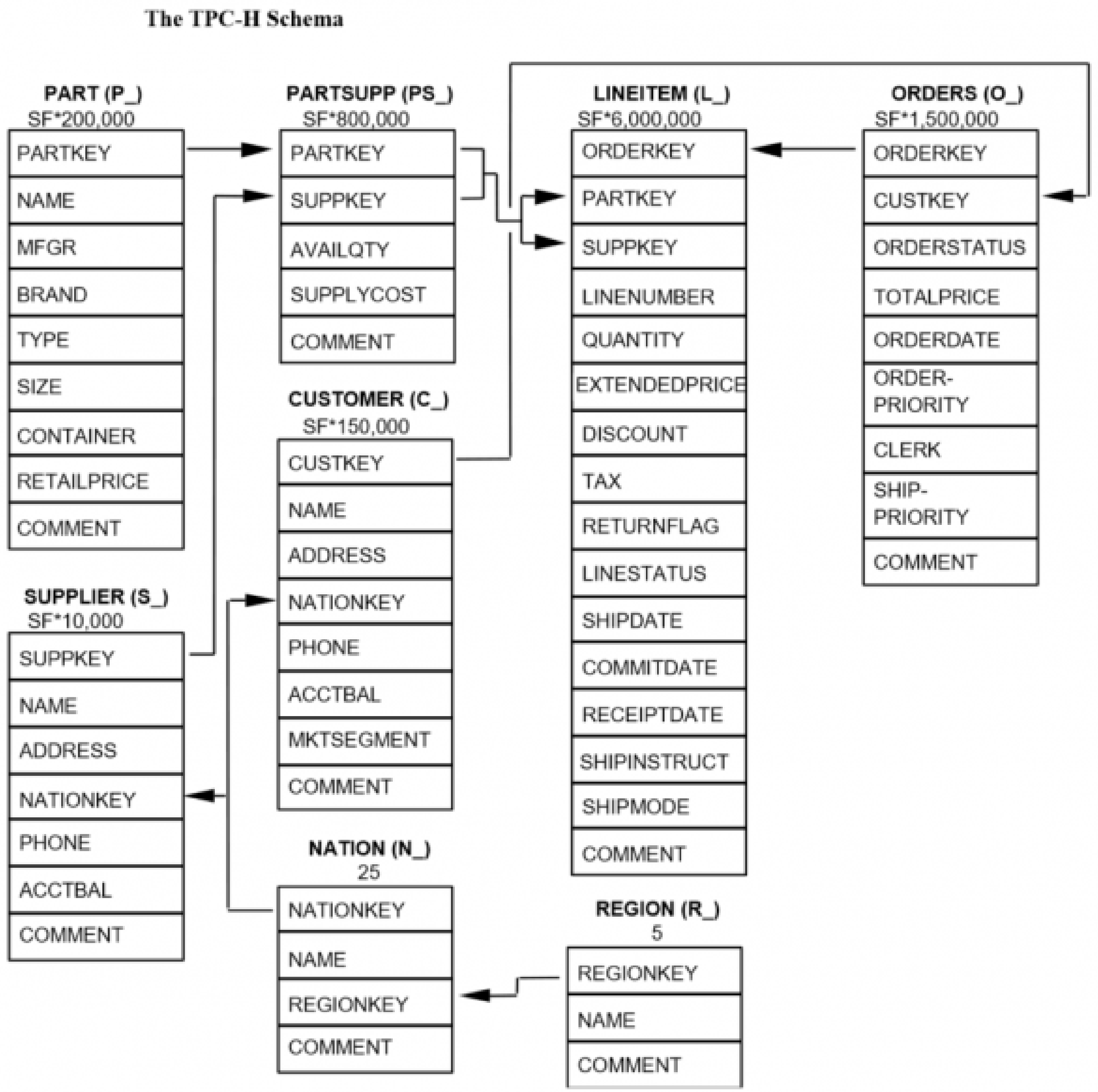
Member

uid	*gid*
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

What is a Database?

A database is a collection of ***relations*** *(or* ***tables****)*

Example 3



Foreign Keys, Referential Integrity

- *Foreign Key: Set of attributes ‘referring’ to a tuple in another relation.*
  - *Must correspond to the primary key of the other relation*
  - *Like a ‘logical pointer’*
- *Foreign key constraints enforce referential integrity (i.e., no dangling references)*

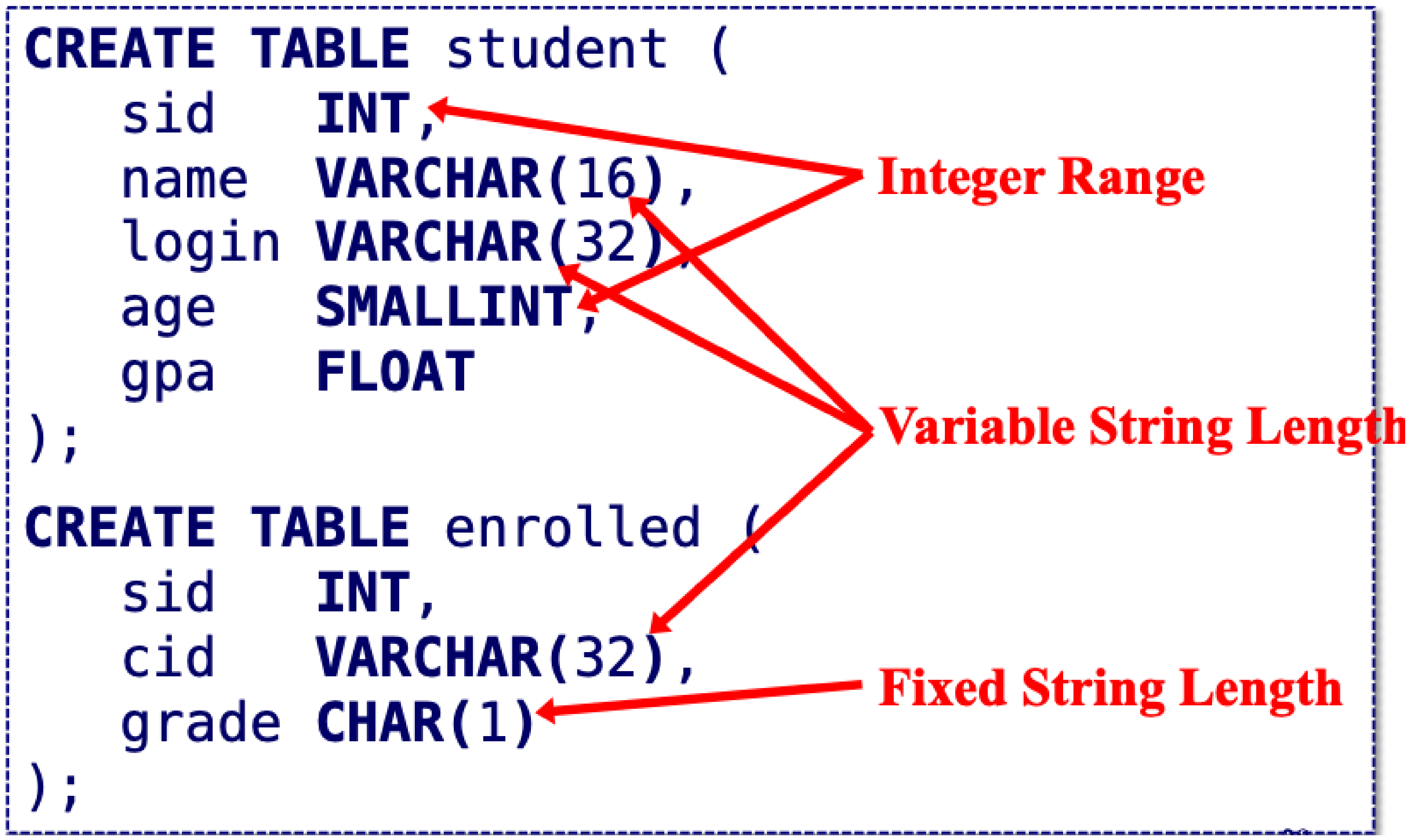
Data Definition Language (DDL)

Data Definition Language (DDL)

```
CREATE TABLE <table-name>(
 [column-definition]*
 [constraints]*
) [table-options];
```

- ***Column-Definition:*** *Comma separated list of column names with types.*
- ***Constraints:*** *Primary key, foreign key, and other meta-data attributes of columns.*
- ***Table-Options:*** *DBMS-specific options for the table (not SQL-92).*

Data Definition Language (DDL)



```
CREATE TABLE student (
    sid   INT,
    name  VARCHAR(16),
    login VARCHAR(32),
    age   SMALLINT,
    gpa   FLOAT
);
CREATE TABLE enrolled (
    sid   INT,
    cid   VARCHAR(32),
    grade CHAR(1)
);
```

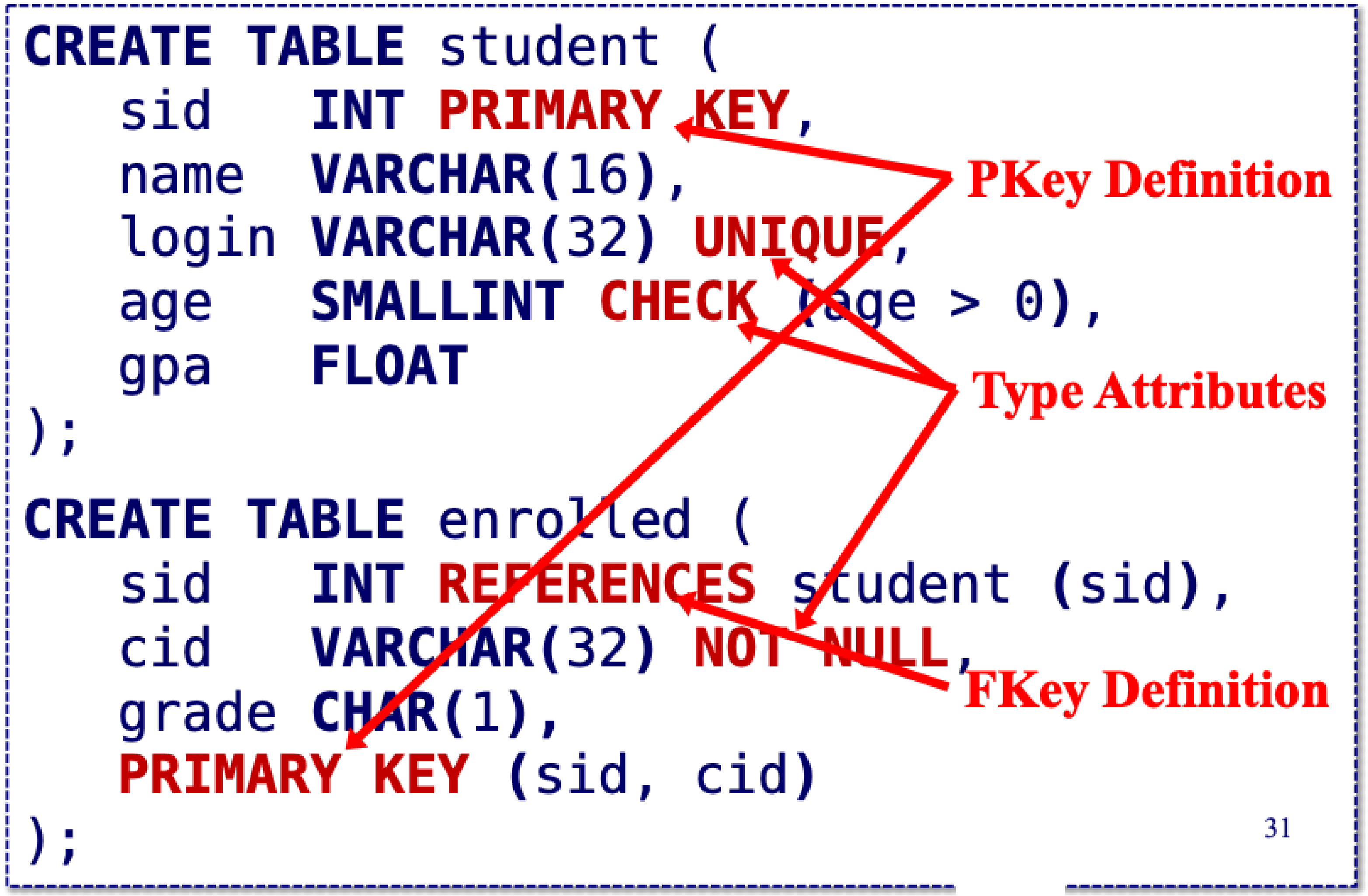
Common Data Types

- *CHAR(n) (fixed length), VARCHAR(n) (variable length)*
- *TINYINT (1 byte), SMALLINT (2 bytes), INT (4 bytes), , BIGINT (8 bytes),*
- *NUMERIC(p,d) (precision (no. of digits) and scale (digits to right of the decimal point), FLOAT, DOUBLE, REAL*
- *DATE, TIME*
- *BINARY(n), VARBINARY(n), BLOB (image, video, files)*

Useful Non-standard Types

- *TEXT*
- *BOOLEAN*
- *ARRAY*
- *Geometric primitives*
- *XML/JSON*
- *Some systems also support user-defined types.*

Integrity Constraints



```
CREATE TABLE student (
    sid   INT PRIMARY KEY,
    name  VARCHAR(16),
    login VARCHAR(32) UNIQUE,
    age   SMALLINT CHECK (age > 0),
    gpa   FLOAT
);

CREATE TABLE enrolled (
    sid   INT REFERENCES student (sid),
    cid   VARCHAR(32) NOT NULL,
    grade CHAR(1),
    PRIMARY KEY (sid, cid)
);
```

PKey Definition
Type Attributes
FKey Definition
31

Primary Keys

- *Single-column primary key:*

```
CREATE TABLE student (
    sid   INT PRIMARY KEY,
    ⋮
```

- *Multi-column primary key:*

```
CREATE TABLE enrolled (
    ⋮
    PRIMARY KEY (sid, cid)
```

Key declarations are part of the schema

```
CREATE TABLE enrolled
  (sid CHAR(20),
   cid CHAR(20),
   grade CHAR(2),
   PRIMARY KEY (sid, cid))
```

PRIMARY KEY == UNIQUE, NOT NULL

Foreign Key References

- *Single-column reference:*

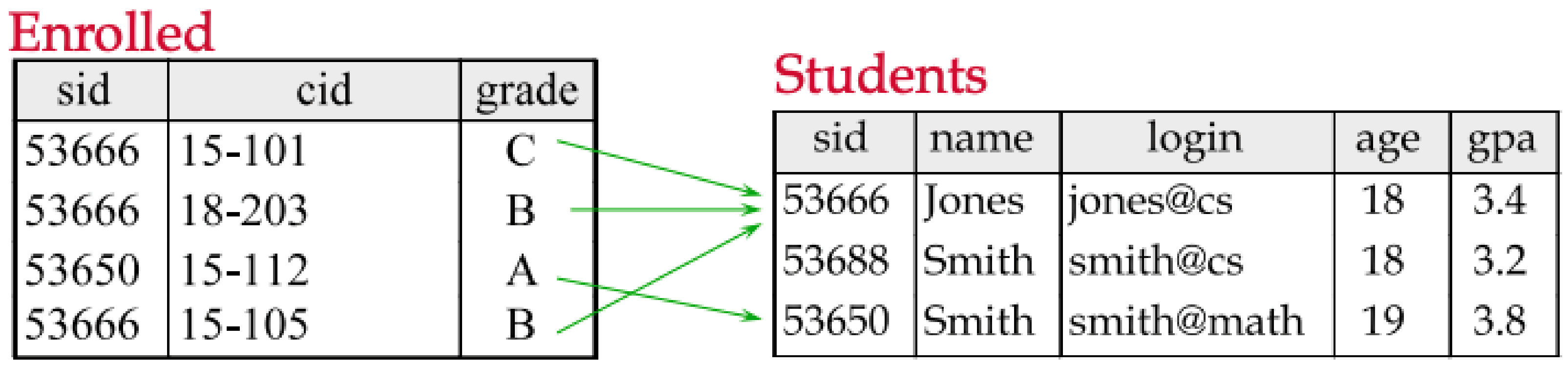
```
CREATE TABLE enrolled (
   sid   INT REFERENCES student (sid),
   ⋮
```

- *Multi-column reference:*

```
CREATE TABLE enrolled (
   ⋮
   FOREIGN KEY (sid, ...)
     REFERENCES student (sid, ...)
```

Foreign Keys in SQL

```
CREATE TABLE Enrolled (
    sid CHAR(20),
    cid CHAR(20),
    grade CHAR(2),
    PRIMARY KEY (sid,cid),
    FOREIGN KEY (sid)
    REFERENCES Students )
```



Enrolled

sid	cid	grade
53666	15-101	C
53666	18-203	B
53650	15-112	A
53666	15-105	B

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@cs	18	3.2
53650	Smith	smith@math	19	3.8

Data Manipulation Language
Basic SQL

Basic Queries: SFW statements

- *SELECT $A_1, A_2, \ldots, A_n$*
  FROM $R_1, R_2, \ldots, R_m$
  WHERE condition;
- *Also called as SPJ (selection-projection-join) query - (selecting rows, projecting columns, and joining tables)* *Select (σ), Project (π), Join (⋈)*
- *Can correspond to (but not exactly equivalent to) relational algebra query:*

$$\pi_{A_1,A_2,\ldots,A_n}\left(\sigma_{condition}\left(R_1 \times R_2 \times \cdots \times R_m\right)\right)$$

Example Database

User

uid	name	age	pop
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3
…	…	…	…

Group

gid	name
abc	Book Club
gov	Student Government
dps	Dead Putting Society
…	…

Member

uid	gid
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

Example: reading a table

- *SELECT * FROM User;*
  - *Single-table query, so no cross product here*
  - *WHERE clause is optional*
  - ** is a short hand for "all columns"*

User

uid	name	age	pop
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3
…	…	…	…

Example: selection and projection

- *Name of users under 18*
  - *SELECT name*
    FROM User
    WHERE age < 18
- *When was Lisa born?*
  - *SELECT 2025 - age*
    FROM User
    WHERE name = 'Lisa';
  - *SELECT list can contain expressions*
    - *Can also use built-in functions such as SUBSTR, ABS, etc.*
  - *String literals (case sensitive) are enclosed in single quotes*

User

uid	name	age	pop
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3
…	…	…	…

Example: join

- *ID's and names of groups with a user whose name contains "Simpson"*

User

uid	name	age	pop
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3

Group

gid	name
abc	Book Club
gov	Student Government
dps	Dead Putting Society
…	…

Member

uid	gid
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

Example: join

User

uid	name	age	pop
142	Bart	10	0.9
123	Milhouse	10	0.2
857	Lisa	8	0.7
456	Ralph	8	0.3
…	…	…	…

Group

gid	name
abc	Book Club
gov	Student Government
dps	Dead Putting Society
…	…

- *ID's and names of groups with a user whose name contains "Simpson"*
  - *SELECT Group.gid, Group.name*
    FROM User, Member, Group
    WHERE User.uid = Member.uid
    AND Member.gid = Group.gid
    AND User.name LIKE '%Simpson%';
  - *LIKE matches a string against a pattern*
  - *% matches any sequence of zero or more characters*
  - *Okay to omit table_name in table_name.column_name if column_name is unique*

Member

uid	gid
142	dps
123	gov
857	abc
857	gov
456	abc
456	gov
…	…

Check out TPC-H on PgAdmin