**CIS 353 Notes**

**Week 1**

Database – Collection of related data

Example: Student – name, year, ID, Major

Mini-World – Part of the real world about which data is stored in a database.

Why Databases?

* To allow management, storage, organize data, security, data comparison

File System: data integrity:

* For CSV’s there is no grantee that there is not overlap or override of data

File System: Implementation:

* CSV cannot ensure that there is proper implementation of the file. They cannot nmake sure more than one person can write to the file at the same time.

File System: Durability:

* What if the machine crashes while our program  
  is updating a record?
* What if we want to replicate the database on  
  multiple machines for high availability?
* CSV’s is not much security in the file system

Database Approach

* Superior to the file approach in terms of efficiency,  
  consistency and maintenance  
  Loose coupling between applications and data  
  Facilities provided for data querying and retrieve

A diagram of a software system

Description automatically generated

The Relational Model:

Must have three basic parts -

Structure: Table/Relation \* assume it is an 2D array

Operations: Relational algebra. These are mechanisms that allow use to query the table. Allow us to retrieve data the way we want.

Constraints: EX. All inputs in field must be a single letter

Relations:

* A relation is a two-dimensional table
  + relation – Table
  + Attribute – Column
  + Tuple – row (not the header row)
* Database – Collection relations

Schema:

* Chema od a relations is the name of the relations fallowed by a parenthesized list of attributes Ex. CourseTaken(Student, Corse, Grade)

Relations and Schema:

Relations is a set up tuples.

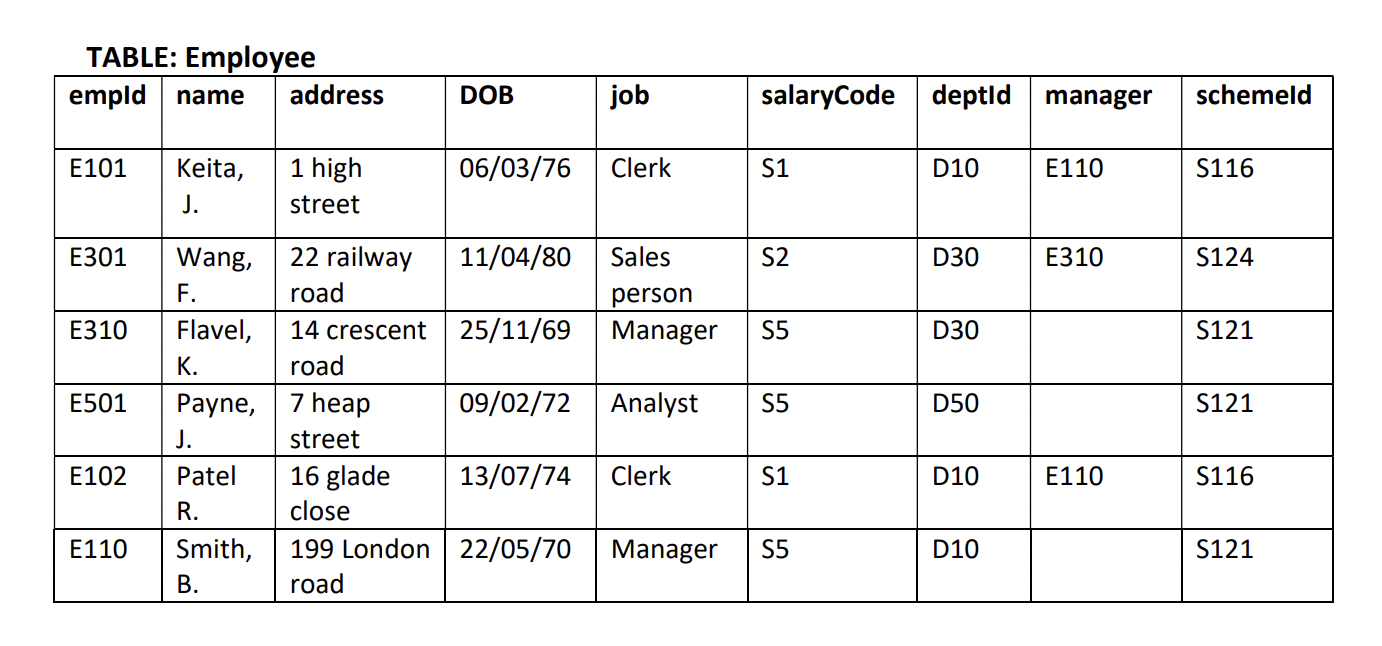
Example: Relation name is Courses Taken that consists of Attributes student, Course and grade.

And also how many student rows are the tuples.

Degree is 3(based on number of attributes)

|  |  |  |
| --- | --- | --- |
| Student | Course | Grade |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Key of Relations:



The goal of the **key** is to uniquely identify something

* You can use more than one keys

Uniquely identifying each tuple

Super > Candidate > Primary

* Super Key
  + Uniquely identifying a tuple
  + May contain a NULL value
  + Super key may contain additional attributes, more than one
  + Largest safe of attributes that can help identity a tuple
  + {ID, Name, address, DOB}
* Candidate Keys
  + Subset of super key
  + The lowest number of attributes needed to uniquely identify the attributes.
  + {ID}, {SSN}, {Email}: If ID is a primary attribute and email is not then {ID, Email}, is not minimal meaning it is not a candidate key
  + If your name and email cannot uniquely identify the attributes on their own but they can together than, {name, email} would be a candidate key. Even if at the time they can be is has to be future proof pair.
* Primary Key
  + Candidate Key with no NULL value
  + Generally, never changed
  + Only one primary key for each relation. There might be more than one possibly primary key but only one can be chosen.
* Alternate Key
  + Candidate keys except the primary key
* Foreign Keys
  + Attribute that Is common between two tables ca be used as a foreign key to link them together.
  + If you need more than one more foreign key to access the right data than that is acceptable.

Relational Constraints:

A screenshot of a computer

Description automatically generated

Example Relational Data Model

A table with numbers and a list of products

Description automatically generated with medium confidence