Review of Database Terms and Concepts

- A **database table** contains data about objects of a particular type or entity. Examples of entities are Customers, Products.
  - The **rows** in a table are called **records.**
  - The **columns** in a table are called **fields** or **attributes**.
  - Columns may have **constraints**. For example, when you define a column as being of data type "Date" (and it is a required field), this means that the database has a constraint that says, no record may be added unless it has a valid date in that field. If you specify another column as being 20 characters, the database will not allow a record to be added if it has more than 20 characters in that field.
  - Each table should have a **primary key**, which is one or more fields that uniquely identifies a record within the table. When you specify a primary key, you are setting up a constraint that says this table "shall not contain more than one record with the same primary key value as another record (in the same table)".
  - A table may have zero or more **foreign keys** which "point to" another record (in the same or different table). When you specify a foreign key (with "integrity enforced"), you are setting up a constraint that says "no foreign key value can point to a non-existing record".
- A **database** is a collection of related tables. When we design a database, we determine which tables need to be included for a given application, what fields or attributes are needed for each table, and how those tables should be related.
- A **database management system (DBMS)** is system software that encapsulates the database. There is no way to add, edit, delete, or even view the data unless you go through the database management system. You can think of the DBMS as the "bouncer" at a night club. Nobody unauthorized gets in. No actions (e.g., updates) are allowed if they would break the rules (constraints).

As we specify the field types and specific relationships between tables, this creates **constraints** to be enforced by the Data base management system (e.g., MSAcess). The DBMS ensures that

- only valid data goes in (e.g., valid dates in date fields, valid numbers in number fields),
- there are no duplicate primary key values in any table, and
- there are no invalid foreign key values (one record pointing to another record that does not exist).

The example below shows

- a **database** with two tables: an Employee **table** and a Department table.
- The Employee table contains 7 **rows** (also called **records** or **objects**), each representing a particular Employee.
- There are 8 **columns** in the Employee table (columns are also called **fields** or **attributes**) – so, that means we know 8 pieces of data (first name, last name, address, etc.) about each Employee in our table.
- In our example, Emp_ID has been specified as the **primary key**. Once a certain field (or fields) is specified as primary key, the DBMS will never allow two records with the same primary key value to be stored in a table.
- In our example, the Employee table has one **foreign key** field, Dept_ID that points to a record in the Department table. Once we tell the DBMS that a foreign key of one table points to the primary key of another table, the DBMS will ensure that there is never a pointer to nowhere.

Employee Table

Emp_ID	Last_name	First_name	Employee_Adrs	City	State	zip_Code	Dept_ID
1	Paquette	Joe	2224 Web Blvd	Centralia	WA	99154	6
2	Johnson	Fred	410 Deer Ave	Centralia	WA	99155	1
4	Bradish	Emily	512 Plaza Place	Seattle	WA	99151	5
5	Braak	John	112 N Lincoln	Centralia	WA	99154	6
6	Shuster	George	5003 Country Ln	Centralia	WA	99155	1
7	Tate	Frank	603 Pinecrest Rd	Chehalis	WA	99148	5

Department Table

Dept_ID	Dept_name	Dept_Expense_Act
1	Marketing	122897
2	Sales	453876
5	Service	677856
6	Manufacturing	187689

Given the above data (with primary keys and foreign keys specified), the DBMS will prevent the following actions.

- Cannot add this record into the Employee table (Duplicate Primary Key). Emp_Id = 7 is already there.

Emp_ID	Last_name	First_name	Employee_Adrs	City	State	zip_Code	Dept_ID
7	Smith	Mary	123 Any St	Chehalis	WA	99148	5

- Cannot add this record into the Employee table (Invalid Foreign Key). There is no Department with Dept_Id = 888

Emp_ID	Last_name	First_name	Employee_Adrs	City	State	zip_Code	Dept_ID
8	Smith	Mary	123 Any St	Chehalis	WA	99148	888

- Cannot delete (would create invalid foreign key for several Employee Records that have Dept_ID = 5)

Dept_ID	Dept_name	Dept_Expense_Act
5	Service	677856

Using a Database concept called "JOIN", you can extract data that LOOKS redundant data (see below), but there is no actual redundancy within the database (no copies of data stored).

```
SELECT Last_name, First_Name, Employee_Adrs, Employee.Dept_Id, Dept_Name
FROM Employee, Department WHERE Employee.Dept_ID = Department.Dept_ID
```

Last_name	First_name	Employee_Adrs	Dept_ID	Dept_Name
Paquette	Joe	2224 Web Blvd	6	Manufacturing
Johnson	Fred	410 Deer Ave	1	Marketing
Bradish	Emily	512 Plaza Place	5	Seattle
Braak	John	112 N Lincoln	6	Manufacturing
Shuster	George	5003 Country Ln	1	Marketing
Tate	Frank	603 Pinecrest Rd	5	Chehalis