Primary Key: A primary key is a constraint in a table which uniquely identifies each row/record in a database table. Primary keys must contain unique values. A primary key column cannot have NULL values. A table can have only one primary key, which may consist of single or multiple fields.

Unique Key: A Unique key is a constraint in a table which identifies each row/record in a database table. An attribute with UNIQUE KEY can have NULL value.

Foreign Key: A foreign key is a field (or collection of fields) in one table that uniquely identifies a row of another table.

Candidate Key: A candidate key is a column, or set of columns, in a table that can uniquely identify any database record without referring to any other data. Each table may have one or more candidate keys, but one candidate key is special, and it is called the primary key.

Fundamentals of Database

1) Field: The smallest piece of meaning full information in a file is called a data item or field.

Example:

<table>
<thead>
<tr>
<th>Name</th>
<th>Locality</th>
<th>City</th>
<th>State</th>
<th>Pin code</th>
</tr>
</thead>
</table>

2) Data: Data are the raw facts, faces of information that represents qualitative or quantitative attributes of a variable.

Example: Height of mount Everest.
3) Information: Set of data that convey a meaning.


4) Knowledge: A knowledge base (KB) is a technology used to store complex structured and unstructured information used by a computer system.

5) Meta data: It is a data about other data. It provides information about other data managed within an application or environment.

6) Database: It is a well defined collection of information that is organised so that it is easily manageable.

7) Database system: A system designed to manage large bodies of information. The DBS must provide safety.

8) Database management system: It is a set of computer programs that controls the creation, maintenance, and the use of database of an organisation and its end user.

**DBMS Concepts:**

1) DBMS Engine: It accepts logical inputs from various other DBMS sub systems and converts them into physical equivalent and actually accesses the database and data dictionary* as they exist on a storage device.

   *Data Dictionary: It is known as the system catalog, it is the repository of all meta data relevant to the object and also of information concerning the
DBMS itself. A data dictionary may contain metadata, security and structures.

2) Data definition subsystem:

1) Data Definition Language (DDL): It helps user to create and maintain the data dictionary and define the structure of the files in a database.

2) Data Manipulation Language (DML): It helps user to add, change and delete information in a database and query it.

3) Application generation: It contains facilities to help to develop transactions interactive application. It usually requires that user performs a detail series of tasks to perform a transaction.

4) Data administrator subsystem: It helps user to manage the overall database by providing facilities of backup and recovery, security management, concurrency control and change management.

**ACID** /*Needs to be revisited*/

1) Data redundancy and inconsistency: Since, different programmers create the files and application programs over a long period the various files likely to have different structures and programs may be written in several programming languages. Moreover, the same information may be duplicated. For example: Address and telephone number of a customer may appear in a file that consists of checking account records. This redundancy, leads to a higher storage and excess cost.

2) Data isolation: Because data are scattered in various files of different formats, writing new applications programs to retrieve the appropriate
3) Integrity: The data values stored in database must satisfy certain types of consistency constraints.
4) Atomicity: One transaction is either successful or failed.
5) Concurrent access anomalies: Reads more than one user simultaneously.

**Types of Users**

1) End user: A person who has no knowledge about the presence and structure of the database.
2) Online user: A person who may communicate with the database directly via an online terminal or indirectly via a user interface ad applications program. These user are aware of database.
3) Application programmers: Professional programmer are those who are responsible for developing applications/UI the application could be written in a general purpose language.
4) Database administrator: It is a person knowledgeable person who is responsible for the physical design and management of database and evaluation and implementation of database (Physical).

**Data abstraction**

The major purpose of a database system is to provide users with an abstract view; the system hides certain features about how/where data is stored. Example: A banker can view details of customers.

**Levels of data abstractions:**
**Internal levels** of DBMS is the one which is closest to data storage. It refers to how data is actually stored in the physical level such as Hard Disk. It is also known as the physical level.

**Logical levels** deal with what data are stored and relationships between data items… /*Needs revisit*/

**View level** is the one which is close to the user. A view involves only data which are only concerned to the user.

**Difference between DA and DBA**

DA is a person who oversees the data integrity of the database, they do not create objects and may not have privileges other than view the data and report any data discrepancy.

Responsibilities: Logical design of database, perform business requirement gathering, analysing requirement, conduct data definition sessions with clients, Assists DBA in creating physical table from logical level.
DBA is a person who administrates the entire database. It can check data in table to make sure there’s no error or to see if the table is right or wrong. It is usually busy with fixing problems, creating table and looking for errors.

Responsibilities: Physical design of DB, Analysis data volume and space requirement, Execute DB backup and recovery, Monitor DB space requirement, verify integrity of data in DB.

**Difference between DBMS and RDBMS**

<table>
<thead>
<tr>
<th>DBMS</th>
<th>RDBMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBMS does not spot client/server architecture.</td>
<td>It spots client/server architecture.</td>
</tr>
<tr>
<td>It does not follow the normalization of DB</td>
<td>It follows normalization of DB.</td>
</tr>
<tr>
<td>It does not allow data to be shared from servers to client</td>
<td>It allows data to be shared from servers to client</td>
</tr>
<tr>
<td>Only one user can access the DB at a time.</td>
<td>It allows simultaneous access of users to the DB</td>
</tr>
<tr>
<td>It doesn’t follow codd’s rules,</td>
<td>It follows codd’s rule</td>
</tr>
<tr>
<td>It is a DB management system</td>
<td>It is a relational DBMS</td>
</tr>
<tr>
<td>It does not join the tables</td>
<td>It can join more than one tables</td>
</tr>
<tr>
<td>It does not use the foreign key</td>
<td>It uses foreign key</td>
</tr>
</tbody>
</table>

**Data Models:**

They are used to design a database. Data models are a collection of conceptual tools for describing data, data relationship, data semantics and consistency.
constraints. It provides a way to design the database at physical, logical and view level. They can be classified into four categories.

1) Relational Model: This model uses a collection of tables to represent both data and relationships among those data. Each table has multiple columns and each column has a unique name. It is a record based model because database is stored in a fixed format.

2) Entity Relationship Data Model: It is based on a perception of a real world that consists of a collection of basic objects (called entity) and of relationship among these objects. An entity is a thing or object that is distinguishable from other tables.

   Example:

3) Object Based Data Model: The object oriented data model is another data model that has been seen increasing attention. Here...

4) Semi Structured Data Model: It permits the specification of data where individual data items of the same type may have different types of attributes. Example: XML is widely used to represent semi structured data models.
**Database languages:**

1) Data Definition Language (DDL): It is used to describe data and data structure of a database with the help of data scheme and also changed later. Example: Typical DDL operations with their respective keywords in SQL. There are various commands of DDL: CREATE TABLE, ALTER TABLE and DROP TABLE.

2) Data Manipulation Language (DML): It is used to store, search and read. Such operations can be done with a data manipulation language with INSERT, MODIFY, UPDATE, DELETE, SELECT.

3) Data Control Model (DCL): It is used to control access to data stored in a data. Example: GRANT, REVOKE.

4) Transaction Control Language (TCL): Transaction Control Language are used to manage the changes made by DML. It allows statements to be grouped into logical transaction. Example: COMMIT, SAVEPOINT, ROLLBACK.

**Difference between DELETE and TRUNCATE:**

<table>
<thead>
<tr>
<th>TRUNCATE</th>
<th>DELETE</th>
</tr>
</thead>
<tbody>
<tr>
<td>It does not allow to specify clause ‘WHERE’.</td>
<td>It does allow to specify ‘WHERE’ clause.</td>
</tr>
<tr>
<td>It removes record by de-allocating memory used by data.</td>
<td>Removes data at logical level.</td>
</tr>
<tr>
<td>Record deleted cannot be rolled backed until data server supports it.</td>
<td>Data deletion can be roll backed.</td>
</tr>
<tr>
<td>It removes all records at once (Like DROP).</td>
<td>It removes the specified records at a time or all record at once.</td>
</tr>
<tr>
<td>It is faster than DELETE command</td>
<td>It is slower than TRUNCATE command.</td>
</tr>
<tr>
<td>It is a part of DDL.</td>
<td>It is a part of DML.</td>
</tr>
</tbody>
</table>

**Keys:**
A key is a single attribute or a combination of two or more attributes of an entity set. This is used to identify instances of the set.

1) Primary Key: A primary key is a field that is uniquely identified in a table as it uniquely identifies each entity. It cannot have NULL or duplicate values. Example: E_No, Voter_Id, Pan_No, adhar_no.

2) Candidate Key: A nominee for primary key field is known as candidate key.

3) Alternate Key: A candidate key which is not a part of the primary key is called the alternate key. Example: If E_no is PK then Voter_ID, Pad_ID and adhaar_no are AK.

4) Composite Key: Creating more than one PK jointly is known as CK. Example: Employee_ID and Voter_ID.

5) Super Key: If we add additional attributes to the resulting combination instance. Such augmented keys are called super key. A PK is called a minimum super key. Example: E_No and name.

6) Foreign Key: It is a PK of master table which is referenced in the current table. So, it is known as foreign key in the current table. It has one or more columns whose value must exist in PK of other tables.

**Entity Relationship Diagram Convention:**

Entity: They are represented by a rectangular box with the name of the entity in the box.

Example: ![Student Entity](image)

An Attribute is shown as an ellipse attached to a relevant entity with a line.
. The entity name is written in uppercase whereas attribute name is written in lower case.

. The PK or key attributes are underlined.

. The attributes are connected using lines. If the attribute is simple single valued, a single line is used.

. If the attribute is a derived attribute then a dotted ellipse is used.

(number_of_students)
Relationship: The association or relationship that exists between entities relates data items in a meaningful way.

Degree of relationship: The degree of a relationship indicates the number of associate entities.

There are four types of relationships: Unary, Binary, Tertiary, and Quaternary.

1) Unary: Exists when an association is maintained within a single entity.

2) Binary: Exists when two entities are associated.
3) Tertiary: When three entities are associated.

4) Quaternary: When there are four entities associated.

**Simple attributes:**
It cannot be divided into simpler components. Example: Age of employee.

**Composite Attribute:**

It can be split into simpler compounds. Example: Name, Data of joining of employee.

**Single Valued attribute:**

It can take only one value for entity instance. Example: Age.

**Multivalued Attribute:**

It can take up many values. Example: Skill set of employee.

**Stored Attributes:**

A stored attribute is an attribute that need to be stored permanently. Example: Name.

**Derived:**

They are derived from another attribute. Example: Year of service from the date of joining.

**Regular Entity and Weak Entity:**

Regular entity has its own key attributes. Example: Emp_ID, E_No.

Weak entity depends on other entities for its existence and does not have any key attributes or its own. Example: Spouse name.

<table>
<thead>
<tr>
<th>Strong/Regular</th>
<th>Weak</th>
</tr>
</thead>
<tbody>
<tr>
<td>An entity set which has a PK.</td>
<td>It does not have sufficient attributes for a PK.</td>
</tr>
<tr>
<td>Example: Entities EMP and STUD are strong entities because they have PK.</td>
<td>They are represented by double rectangles.</td>
</tr>
</tbody>
</table>
**Database Schema:**

It is a description of a database which is specified during database design and it does not change very often.

Example: STUDENT

<table>
<thead>
<tr>
<th>Name</th>
<th>Student_ENo</th>
<th>Class</th>
<th>Major</th>
</tr>
</thead>
</table>

A Schema diagram is often used to represent displaying the schemas. The data is the database at a particular moment in time is called DB.

**State/Instance:** A database state is also called instance change everytime data is inserted, deleted or modified.

**Database Schema VS Database State:**

When we define a new database, we specify the database schema only to DBMS or current state of the database is the empty with no data. We get the initial state of the database when the database is loaded with initial data.

**Referential Integrity:**

In a Relation database we want to ensure that a value that appears in one relation for a given set of attributes also appears for a certain set of attributes in another table. This is called referential integrity. It is concept of relation based on the concept of PK or FK.
The following table follows Referential Integrity:

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Child table</td>
<td>A table where referential constraints are defined.</td>
</tr>
<tr>
<td>Parent table</td>
<td>Table being referenced by a child table</td>
</tr>
<tr>
<td>PK</td>
<td>A PK uniquely identifies a row in a table.</td>
</tr>
<tr>
<td>FK</td>
<td>Foreign key refers a column in the child table. A FK may contains upto 16 columns.</td>
</tr>
<tr>
<td>Referencing Columns</td>
<td>It is written in a referencing table that are FK for a column in other referencing column</td>
</tr>
<tr>
<td>Referenced Column</td>
<td>PK column or unique column in a referenced table.</td>
</tr>
</tbody>
</table>

**Generalization:**

It is the result of taking the union of several lower level entity sets to procedure to higher level entity set. There are similarities between SAVING_ACCOUNT and CURRENT_ACCOUNT.

Account entity set in a bank account. These similarities are in the sense that they have several attributes in common. This can be expressed by generalization. Which is containment relationship that exists between a higher level entity set and lower level entity set? Example: Account is a higher level entity and saving and current account are lower level.
Aggregation:

Process of compiling information on an object. Thereby, abstracting a Higher level object or entity set.
Specialization:
It is the opposite to generalization. It is a top – down approaches in which, one higher level entity can be broken down into two lower level entity.