DATA MODELING IN CASSANDRA:

Data modeling is the process of identifying entities and their relationships. In relational databases, data is placed in normalized tables with foreign keys used to reference related data in other tables. Queries that the application will make are driven by the structure of the tables and related data are queried as table joins.

**CONCEPTUAL DATA MODELING IN CASSANDRA :**

Let us create a domain model which can be understand easily in the relational data base world and we can trace the mapping , from the relational to distributed hashtable in Cassandra .



This is an example of an ER diagram for a basic employee management system's conceptual data model. This organization has numerous departments, each with a large number of personnel. Workers can only work on one project at a time and are assigned to several projects; some workers may not have been assigned to any projects at all. Every employee has a position, or a job role; roles can be shared by multiple employees. There might also be open positions. To access the company's shared network, each employee has a login, however their user capabilities vary.

In order to fulfill these business needs, I created the conceptual data model that is displayed above. Employee, Department, Project, Role, and Login are the five entities that it has, and they all correspond to the actual entity data found in the physical database. The business scenario depicted is supported by the following four relationships:

**Department\_Employee:** A one-mandatory-to-many-mandatory relationship.

**Project\_Employee:** A one-optional-to-many-mandatory relationship.

**Role\_Employee:** A one-mandatory-to-many-optional relationship.

**Employee\_Login:** A one-mandatory-to-one-mandatory relationship.



**DEFINING APPLICATION QUERIES :**

* Find the basic required employee details
* Finding the job dept of employee
* Finding the qualifications of employee
* Finding the salary of employee
* Finding the leaves taken by employee

All of the queries are shown in the context of the workflow of the application in the figure below. Each box on the diagram represents a step in the application workflow, with arrows indicating the flows between steps and the associated query. If you’ve modeled the application well, each step of the workflow accomplishes a task that “unlocks” subsequent steps.

**Cassandra Data Model Components:**

Cassandra data model provides a mechanism for data storage. The components of Cassandra data model are keyspaces, tables, and columns.

**Keyspaces**

Cassandra data model consists of keyspaces at the highest level. Keyspaces are the containers of data, similar to the schema or database in a relational database. Typically, keyspaces contain many tables.

Since there is no default keyspace, one must establish one before creating tables.

A table is just a part of one keyspace, and a keyspace can contain any number of tables. This illustrates a link between one and many.

Keyspace specifications govern replication. Replication of three, for instance, indicates that there will be three copies of each data row in the keyspace.

Additionally, when creating keyspace, you must specify the replication factor. Later on, though, the replication factor can be changed.



**Tables**

Within the keyspaces, the tables are defined. Tables are also referred to as Column Families in the earlier versions of Cassandra. Tables contain a set of columns and a primary key, and they store data in a set of rows.

Data is presented in tables in two formats: horizontally as rows and vertically as columns. Among the characteristics of tables are:

Tables consist of numerous rows and columns. As was previously indicated, in previous Cassandra versions, a table was also referred to as a Column Family.

* Certain Cassandra error messages and documentation still refer to it as column family.
* Determining a table's primary key is crucial.

The statement to create a table named employee with four columns—empid, empfirstname, emplastname, and empsalary—is displayed in the query that is provided.

**Columns**

Columns define the structure of data in a table. Each column has an associated type, such as integer, text, double, and Boolean.

Columns consist of various types, such as integer, big integer, text, float, double, and Boolean.

Cassandra also provides collection types such as set, list, and map.

Further, column values have an associated time stamp representing the time of update.

This timestamp can be retrieved using the function writetime.

Using these equipments data modeling is done in Cassandra

You can now start creating Cassandra tables after you have defined your queries. First, take the entities and relationships from the conceptual model and turn them into a logical model with a table for every query.

There are 3 main modeling of the data types

* LOGICAL DATA MODELING
* PHYSICAL DATA MODELING
* EVALUTING AND REFINING DATA MODELING