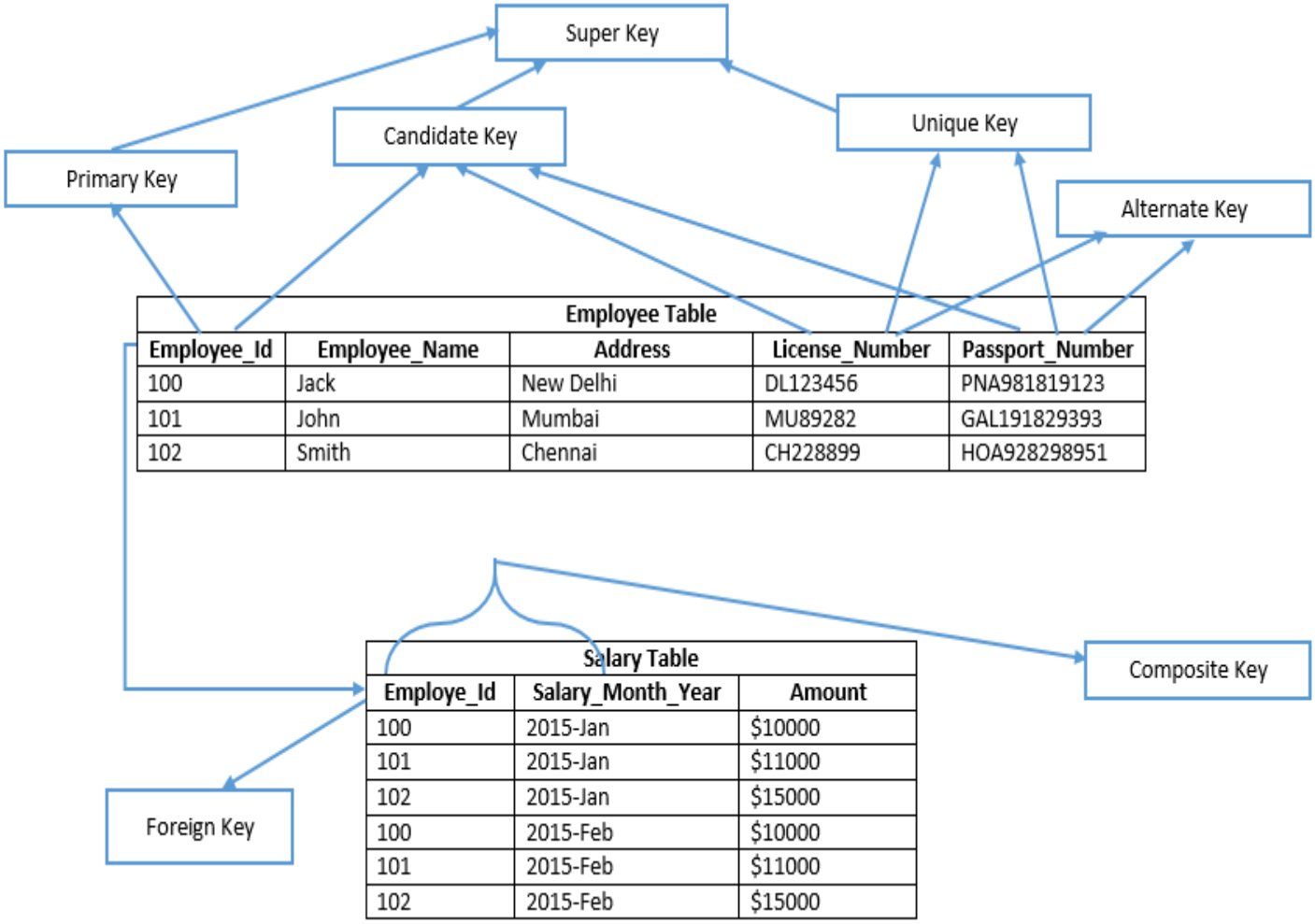
Keys are fields in a table which participate in below activities in RDBMS systems:

1. To create relationships between two tables.
2. To maintain uniqueness in a table.
3. To keep consistent and valid data in database.
4. Might help in fast data retrieval by facilitating indexes on column(s).

SQL Server supports various types of keys, which are listed below:

1. Candidate Key
2. Primary Key
3. Unique Key
4. Alternate Key
5. Composite Key
6. Super Key
7. Foreign Key



Employee Table

Employee_Id	Employee_Name	Address	License_Number	Passport_Number
100	Jack	New Delhi	DL123456	PNA981819123
101	John	Mumbai	MU89282	GAL191829393
102	Smith	Chennai	CH228899	HOA928298951

Salary Table

Employe_Id	Salary_Month_Year	Amount
100	2015-Jan	$10000
101	2015-Jan	$11000
102	2015-Jan	$15000
100	2015-Feb	$10000
101	2015-Feb	$11000
102	2015-Feb	$15000

Candidate Key

Candidate key is a key of a table which can be selected as a primary key of the table. A table can have multiple candidate keys, out of which one can be selected as a primary key.

Example: Employee_Id, License_Number and Passport_Number are candidate keys

Primary Key

Primary key is a candidate key of the table selected to identify each record uniquely in table. Primary key does not allow null value in the column and keeps unique values throughout the column. In above example, Employee_Id is a primary key of Employee table. In SQL Server, by default primary key creates a clustered index on a heap tables (a table which does not have a clustered index is known as a heap table). We can also define a nonclustered primary key on a table by defining the type of index explicitly.

A table can have only one primary key and primary key can be defined in SQL Server using below SQL statements:

1. CRETE TABLE statement (at the time of table creation) – In this case, system defines the name of primary key
2. ALTER TABLE statement (using a primary key constraint) – User defines the name of the primary key

Example: Employee_Id is a primary key of Employee table.

Unique Key

Unique key is similar to primary key and does not allow duplicate values in the column. It has below differences in comparison of primary key:

1. It allows one null value in the column.
2. By default, it creates a nonclustered index on heap tables.

Alternate Key

Alternate key is a candidate key, currently not selected as primary key of the table.

Example: License_Number and Passport_Number are alternate keys.

Composite Key

Composite key (also known as compound key or concatenated key) is a group of two or more columns that identifies each row of a table uniquely. Individual column of composite key might not able to uniquely identify the record. It can be a primary key or candidate key also.

Example: In salary table, Employee_Id and Salary_Month_Year are combined together to identify each row uniquely in Salary table. Independently Employee_Id or Salary_Month_Year column cannot identify each row uniquely. We can create a composite primary key on Salary table using Employee_Id and Salary_Month_Year columns.

Super Key

Super key is a set of columns on which all columns of the table are functionally dependent. It is a set of columns that uniquely identifies each row in a table. Super key may hold some additional columns which are not strictly required to uniquely identify each row. Primary key and candidate keys are minimal super keys or you can say subset of super keys.

In above example, In Employee table, column Employee_Id is sufficient to uniquely identify any row of the table, so that any set of column from Employee table which contains Employee_Id is a super key for Employee Table.

For **example:** {Employee_Id}, {Employee_Id, Employee_Name}, {Employee_Id, Employee_Name, Address} etc.

License_Number and Passport_Number columns can also identify any row of the table uniquely. Any set of column which contains License_Number or Passport_Number or Employee_Id is a super key of the table.

For **example:** {License_Number, Employee_Name, Address}, {License_Number, Employee_Name, Passport_Number}, {Passport_Number, Employee_Name, Address, License_Number}, {Passport_Number, Employee_Name}, {Passport_Number, Employee_Id} etc.

Foreign Key

In a relationship between two tables, a primary key of one table is referred as a foreign key in another table. Foreign key can have duplicate values in it and can also keep null values if column is defined to accept nulls.

Example: Employee_Id (primary key of Employee table) is a foreign key in Salary table.

Natural key.

A key that is formed of attributes that already exist in the real world. For example, U.S. citizens are issued a Social Security Number (SSN) that is unique to them (this isn't guaranteed to be true, but it's pretty darn close in practice). SSN could be used as a natural key, assuming privacy laws allow it, for a *Person* entity (assuming the scope of your organization is limited to the U.S.).

Surrogate key.

A key that has no business meaning. An example of which is the AddressID column of the Address table in Figure 1. Addresses don't have an "easy" natural key because you would need to use all of the columns of the Address table to form a key for itself (you might be able to get away with just the combination of Street and ZipCode depending on your problem domain), therefore introducing a surrogate key is a much better option in this case.

Create Commands

1. **Primary Key :**

A. Create Primary Key

```
2. CREATE TABLE CUSTOMERS(
3.         ID   INT              NOT NULL,
4.         NAME VARCHAR (20)     NOT NULL,
5.         AGE  INT              NOT NULL,
6.         ADDRESS  CHAR (25) ,
7.         SALARY   DECIMAL (18, 2),
8.         PRIMARY KEY (ID)

);
```

B. Add primary key in existing Table

```
ALTER TABLE CUSTOMER ADD PRIMARY KEY (ID);
```

C. Add more than one column as Primary key

```
ALTER TABLE CUSTOMERS
   ADD CONSTRAINT PK_CUSTID PRIMARY KEY (ID, NAME);
```

D. Delete primary key

```
ALTER TABLE CUSTOMERS DROP PRIMARY KEY ;
```

1. **Foreign Primary Key :**

A. Create Key

```
CREATE TABLE ORDERS (
       ID          INT        NOT NULL,
       DATE        DATETIME,
       CUSTOMER_ID INT references CUSTOMERS(ID),
       AMOUNT     double,
       PRIMARY KEY (ID)
);
```

B. Add key in existing Table

```
ALTER TABLE ORDERS
   ADD FOREIGN KEY (Customer_ID) REFERENCES CUSTOMERS (ID);
```

D. Delete key

```
ALTER TABLE ORDERS
   DROP FOREIGN KEY;
```