Date: 02/09/2021 DBMS. 07. Keys.

Written by Abhishek Sharma. from GFG Placement course.

In this, we will talk about different keys on databases. There are various keys that are used in Databases. such as: candidate key, super key and primary key.

candidate key: A candidate key is the minimal set of attributes that derives all other attributes.

Example.

Enroll. No.	Student Name	AAdhar No.	Address	pan. no.	contact.
101	Ram	101---9	ABC	AEJ0---9	62---- 09
102	Ramesh	201---8	BCD	BJK--- 8	62 --- 94
103	Suresh	501---6	EFG	CKS---6	98--- 50
104	Sunil	809---1	GHI	MIK----5	93---- 62
105	Mohan	725---6	IJKL	KLM---4	95----- 68
106	Kamlesh	421---5	MNOS	MNO----6	63----95
107	Ritesh	015---6	PQR	STV----5	98---96

* Name is not a candidate key bcz. Another person with the same name may be there.
* Aadhar No. is a candidate key. bcz. it is unique
* Enroll No. is a " " " " "
* Address is not a cantlidate key. bcz two students they might be brother with the same address.
* Pan no. is a candidate key. (Assuming it is unique).
* Contact may or may not be candidate key. bcz. it is dependent upon business logic. If it is allowed to give phone no. only once then it is a candidate key otherwise not.

So, Enrollment NO, Addhar NO, Pan no. → candidate Key.

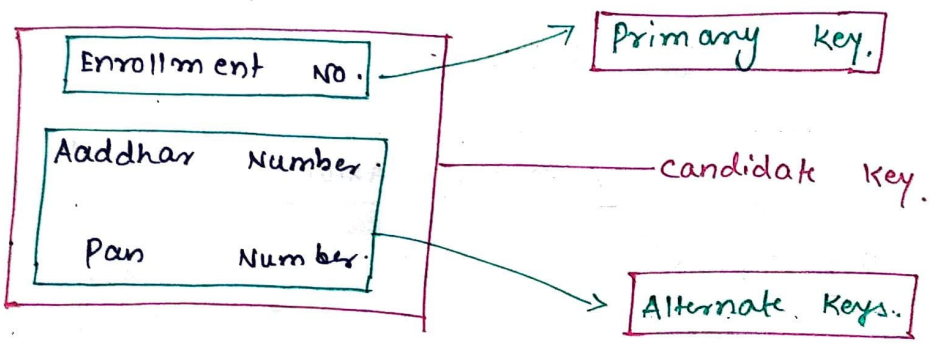
We only pick one out of above candidate key that is called Primary key. (chosen by the Database Designer).

Ex. Enrollment NO. → Primary Key.

Primary Key: The one key that is chosen from multiple candidate key is called Primary Key.

and other keys are called alternate Keys.

Ex.



** candidate key must be distinct and not null. (unique).

Properties ↗

Super key: : Greater than or equal to Super key.

↳ Any set of Attributes that uniquely identifies all the rows is Super key.

Ex. (Enrollment NO, Student Name) :- ↘ This is a key but This can not be a candidate key bcz. candidate Key must be minimal.

It is a Super Key.

* If we minimise Super key, it we remove unnecessary Attributes out of it then it becomes the candidate key.

* Example to find Keys (candidate Key, Super Key, Primary Key)

Ex: 1) R_1 (A, B, C, D)

(A → B means A derives B)

A → B , A → C   C → D

Here A derives B, A derives C and C derives D. and A derives itself.

$A^+ = \{ A\ B\ C\ D \}$ ⇝ clouser of a candidate Key.

That means 'A' is the candidate Key. and A, AB, ABC, AC, and many more are super key.

The Rule that Derives A → C and C → D that means A → D is called Armstrong Axions.

Armstrong Axions.

1) Reflexivity : A Derives A (itself).

2) Transitivity : A → C and C → D then A → D.

3) Augmentation : It says :- if X → Y then XZ → YZ.

These are The Three Armstrong Axions.

using these Three Basic Rules we can find colouser of a candidate key.

Example : 2. R_2 (A, B, C, D)

$AB \rightarrow CD$

Here, AB Derives CD so AB is the candidate key.

Example : 3. R_3 (A, B, C, D)

$B \rightarrow AC$, $C \rightarrow D$

using Reflexivity Rule we can say that B derives A, B, C and using tranivity we can say B derives C. and C derives D so. B derives D.

$B^+ = \{A\ B\ C\ D\}$

So, B is the candidate key.

for super key, we can add anything in B like..

BA, B, BAC, BAD, BD and many more are super keys.

Super keys : $2^n - 1$

$2^4 - 1 = 15$ ↑ in the above 3 example.

* Candidate Keys and super keys does not having NULL value.

that means: Every Table does have atleast one super key Which means atleast one candidate keys.

So, you always have one Super key, one candidate key and one primary key as well.

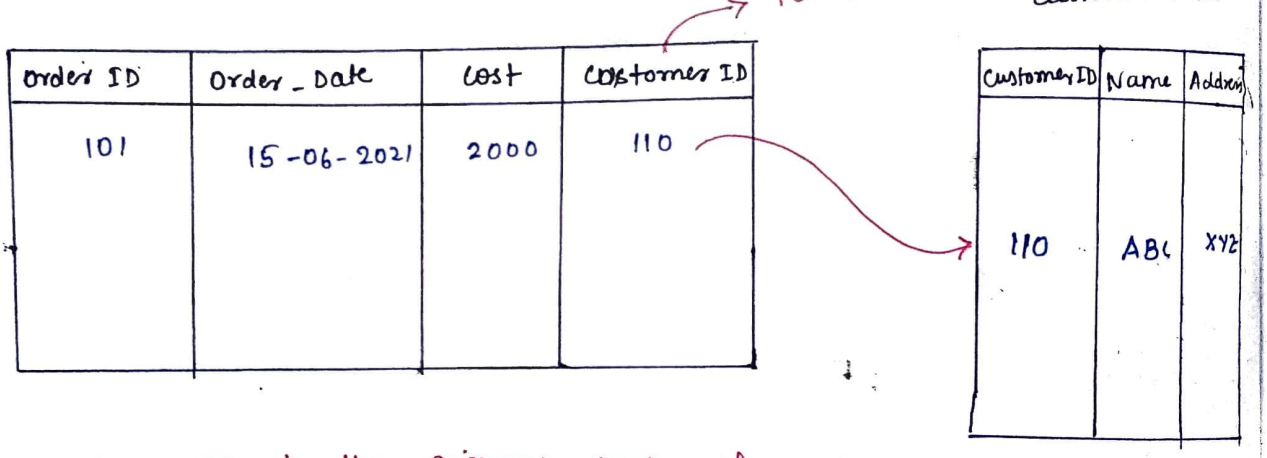
08. Foreign Keys.

Foreign keys refers to primary key of some other table. It may refers primary key of same table. but it most of the time refering to some other table.

It doesn't have to be primary key all time, it might be unique key as well. but we prefer to take primary key.

Example of Foreign keys.

Order Table of a E-commerse website.



* Customer ID is the primary key of order table. So we will took it and refrencing to another table that represents the customer table of that same customer ID.
* This is the function of ~~Primary~~ Foreign Key.
* In code the Example s will be. like this

```
Order Table    Order (
    Order-id    INTEGER Primary key,
    order-Date    Date,
    cost    Integer,
    Customer-ID    REFERENCES    Customer (Customer ID))
```

// ~~Primary~~ Foreign key //

* If we want to delete that information from the customer table (110) then we will use "ON DELETE CASCADE"

their are many ↓ methods that are provided by databases. like default, NULL and many more more.

and this integrity which is maintained by databases (to delete the data if the customer wants) is called Referential Integrity.

---

09. DATABASE Normalisation.

We do Database Normalisation to ensure that data Redundancy is minimum (↓) and Data Integrity is maximum (↑)

DATA
Redundancy : having the same data in multiple places specially string type of data not ID type of DATA.

DATA Integrity : Insuring that their is no arrouneous data inside your Database. , maintainers of data, follows the Busines Rule.

Data Base Normalisation.

Data Redundancy (↓). Low

Data Integrity (↑). high.

* Objectives of Good Database Design.

* No updatation, insertion, deletion anomalies.
  ↳ नियमहिनता (Doesn't follow the rules)
* Early Extendible.
* Good Performence for all query sets.
* More Informative.

1) Updatation Annomaly

If we want to update something in our database table. Updatation Anamoly occures when we have Redundancy in our database.

How we will remove these ??

Ans:- we will split the database table in two or more tables.

Example: 1

Student ID	Student Name	Subject ID	Subject Name.
1001	ABC	CS101	DataBase Management Syst
1002	MNO	CS101	DataBase Management Sys
1001	PQR	CS102	Operating System
1003	ABC	CS102	Operating System
1002	XYZ	CS102	DataBase Managmnt System

If we want to update Here DBMS.

So we have to work more upon the all Query.

Feasible way: Split the table into 2 table and update.

ii) Insertion Anomaly.

We can't insert only student name or student ID if there is already said that you have to fill all the column. This is called insertion Anomaly.

Example. 2

Student ID	Student Name	Stubject ID	Subject Name.
1004	JKL	??	??

We have to fill all the columns. This is called. Insertion Anomaly.

iii) Deletion Anomaly.

If we delete some record but the deleted record which was required to be their even after deletion then it is Deletion Anomaly.

Example: 3

Suppose student 1001 and 1002 Leaves the college and we Deleted both the entries from our Database.

(Example 1)

So, the subject Database Management System also deleted from our record. But we don't want to delete it

This is called Deletion Anomaly.

All these Anomaly which we learnt happens in our Databases bcz. we have Data Redundancy. for get rid of them we need to make seperate tables.

Keep it in mind :- Breaking into tables in more tables we also have caused problems. bcz. if we do this we have many many tables and which is not easy to handle.
⟹ So break the tables but not in too many tables.

10. Functional Dependency.

We studies functional dependency for normalisation of our Databases.

Q) Why we study normalisation ?

Ans: TO Avoid Redundandary in our Databases.

Q) Why we avoid Redundandary in Databases ?

Ans: TO Avoid Anomalies in our Databases like insertion, deletion, updation Anomalies.

Now we will see Functional Dependency.

$$A \longrightarrow B$$

i.e. B is functionally dependent on A.

Example. 1

Enroll No.	Name	Address	phone number

Enroll No. $\longrightarrow$ Name.

Enroll No. $\longrightarrow$ Address

Enroll No. $\longrightarrow$ phone No.

i.e. Name, Address and phone no. is dependent on Enroll no.

Or Enroll No. uniquelly defines Name, Address and phone no.

Enroll No is the Functional Dependency Here.

Example: 2

Subject ID	Student ID	Marks	Grade.

Subject ID, Student ID → Marks.

Subject ID, Student ID → Grade.

Or we can write : Subject ID, Student ID → Marks, Grade.

Only subject ID or Student ID can't define marks bcz.

A student have more than one marks he can be inrolled in many subjects. that why we have to take both.

Here. Subject ID & Student ID is functional Dependency.

Ex: 3

A	B
x	1
y	1
z	2

Here:

A → B ✓✓ (True).

B → A ✗✗ (Not True)

Bcz. if we say '1' it may be x or y.

Functional Dependency.

For a given attribute, if you can uniquely define the other attributes then there is a functional dependency.

Example: 4

Enroll No.	Name	Address
101	Raj	ABC
102	Ravi	ABC
103	Raj	XYZ

Enroll No → Name, Address. ✓ (True)

Name → Address ✗ (False)

Address → Name. ✗ (False)

Q) Why do we study Functional Dependency ??

Soln If we are given a database like this

Student ID	Name	Department ID	Depart. Name.
101	ABC	10	CS
102	BCD	11	ECE
103	ABC	10	CS
104	XYZ	11	ECE
105	CDE	10	CS.

Now from the given above table first we will find the functional dependency. i.e.

Depart. ID ⟶ Depart. Name.

Now we will delete that column (Department Name) (X) and create a seperate table. and link with that table.

Depart - ID	Depart. Name
10	CS
11	ECE

That's How we reduce Avoid. Data Redundancy.

Two Types.

Functional Dependency.

Trivial

Ex. AB → A

A → A

ABC → AC

Attributes that defines itself.

Non Trivial.

Ex. A → B

AB → C

BC → DEA.

Attributes that gives some info

1. First Normal Form (1NF)

Abhishek Sharma Notes.

Rule: Every Attribute should contain only single value (Atomic).

Rule: Every Attribute should contain only single value. i.e. Atomic

Example:

Customer ID	Name	Mobile No.
101	ABC	6290903490, 6291005 70
102	BCD	9598462 7,
103	XYZ	9586756 7,
104	PQR	9354 32 01.

This table is not in INF bcz. as the defⁿ says every Attribute should contain only single value, but the Attribute Mobile No. containing 2 values.

⇓ Conversing the table in 1NF.

1st possible way.

We can add a new column for mobile No. 2. like this

customer ID	Name	Mobile No. 1.	Mobile No. 2
101	ABC	6290903490	6291 00570
102	BCD	9598 46 27	—
103	XYZ.	9586 7526	—
104	PQR	93543201.	—

Theoritically it is in 1NF but the Problem with this table is that there are too many Empty fields.

⇓ Now we will further optimise this table for better 1NF.

Customer ID	Name.	Mob No.
101	ABC	6290903490
102	BCD	95984627
103	XYZ	9586 7526
104	PQR	9394 3201
101	ABC	6291 00570

* Now '101' has two entry. This is in 1NF and Better than previous one bcz. previously we are wasting too much empty spaces. We are repeating Here. in the same table.
This is Better 1NF than Previous one.

* But This table has its own demerit though it is in 1NF. We already know that customer ID should be unique but here it is repeating.

We will Remove this problem in 2NF and 3NF.

As of now this is in 1NF.

⇓

We can modify it via breaking it in two tables. like this

⇓ Better solution

Customer ID.	Name.
101	ABC
102	BCD
103	XYZ
104	PQR

ID.	Customer ID	Mobile No.
1	101	6290903490
2	102	95984627
3	⋮	⋮
⋮	⋮	⋮

* Now customer ID is unique.

Best Idea

We will see further in 2NF & 3NF.