Keys in DBMS (Unit-3)

A key refers to an attribute / a set of attributes that help us identify a row (tuple) uniquely in a table (relation).

It is also used to uniquely identify any record or row of data from the table. It also used to establish and identify relationships between tables.

Types of Keys

⇒ Primary Key
⇒ Candidate Key
⇒ Super key
⇒ Foreign key
⇒ Alternate Key
⇒ Composite key
⇒ ~~Unique~~

Primary Key (Unique + Not Null)

→ It is a unique key
→ It has no duplicate values, it has unique values
→ It cannot be NULL.
→ Primary keys are not necessarily to be a single column, more than one column can also be a primary key for a table.

Note:- There can be more than one candidate key in relation, out of which one can be chosen as the primary key.

for eg: Stud_No, as well as Stud_Phone, are candidate keys for relation Student but Stud_No can be chosen as the primary key (only one out of many candidate keys).

Candidate Key :- The minimal set of attributes that can uniquely identify a tuple is known as a Candidate key. for eg. Stud_No in Student relation.

→ It is a minimal Super key.
→ It is a Super key with no repeated data is called a candidate key.
⇒ The minimal set of attributes that can uniquely identify a record.
⇒ It must contain unique values.
⇒ It can contain NULL values.
⇒ Every table must have at least a single Candidate key.
⇒ A table can have multiple candidate keys but only one primary key.
⇒ The value of the Candidate key is Unique and may be Null for a tuple.
⇒ There can be more than one candidate key in a relationship

Super Key:- The set of attributes that can uniquely identify a tuple is known as Super key.

for eg. Stud_No, (Stud_No, Student_Name), etc.

A super key is a group of single or multiple keys that identifies rows in a table. It supports NULL values.

⇒ Adding zero or more attributes to the candidate key generates the super key.

⇒ A Candidate key is a super key but vice versa is not true.

Eg. Stud_No + Phone is a super key.

Alternate Key:- The Candidate key other than the primary key is called an alternate key.

⇒ All the keys which are not primary keys are called Alternate keys.

⇒ It is a secondry key.

⇒ It contains two or more fields to identify two or more records.

Eg. SName and Address is Alternate keys.

Foreign Keys

⇒ It is a key it acts as a primary key in one table and it acts as secondry key in another table.

⇒ It combines two or more relations at a time.

⇒ They act as a cross-reference between the tables.

Composite Key: Sometimes, a table might not have a single column / attribute that uniquely identify all the records of a table.

To uniquely identify rows of a table, a combination of two or more column/attribute can be used. It still can give duplicate values in rare case. So we need to find the optimal set of attributes that can uniquely identify rows in a table.

⇒ It acts as a primary key if there is no primary key in a table.

⇒ Two or more attributes are used together to make a composite key.

⇒ Different combinations of attributes may give different accuracy in terms of identifying the rows uniquely.

⇒ There can be more

Functional Dependency

The functional dependency is a relationship that exists between two attributes. It typically exists between the primary key and non key attribute within a table.

$$X \rightarrow Y$$

The left side of FD is known as a determinant, the right side of the production is known as a dependent.

For example:

Assume we have an employee table with attributes Emp-Id, Emp-Name, Emp-address.

Here Emp-Id attribute can uniquely identify the Emp-Name attribute of employee table because if we know the Emp-Id, we can tell that employee name associated with it.

Functional Dependency can be written as -

$$\text{Emp-Id} \rightarrow \text{Emp-Name}$$

Types of Functional Dependency

1) Trivial Functional Dependency
2) Non-Trivial Functional Dependency

1) Trivial Functional Dependency

* $A \rightarrow B$ has trivial function dependency if B is a subset of A.
* The following dependencies are also trivial like: $A \rightarrow B$, $B \rightarrow B$

Eg.

Emp-Id	Emp-Name

(Emp-Id, Emp-Name) → Emp-Id

it is a trivial functional dependency as Emp-Id is a subset of (Emp-Id, Emp-Name)

Also, Emp-Id → Emp-Id

and Emp-Name → Emp-Name

are trivial dependencies

rows uniquely.

Non-Trivial Functional Dependency

* $A \rightarrow B$ has a non-trivial functional dependency if B is not a subset of A.

Eg. ID → Name

Name → DOB

Other Functional Dependency

* Multivalued Functional Dependency

In multivalued functional Dependency, entities of the dependent set are not dependent on each other.

i.e. if $\not\exists$ $a \rightarrow \{b, c\}$ and there exists no functional dependency between b and c, then it is called a multivalued functional dependency.

for eg.

roll_no	Name	age

roll_no → {name, age}

It is a multivalued functional dependency, since the dependents name & age are not dependent on each other.

i.e. name → age or age → name doesn't exist.

* Transitive Functional Dependency

In transitive functional dependency, dependent is indirectly dependent on determinant.

i.e. if a → b & b → c then according to axiom of transistivity a → c. This is a transitive functional dependency.

for Eg.

Enroll_No	name	Dept	Building No

Here Enroll_No → Dept
Dept → Building_No

Then according to axiom of Transitivity,
Enroll_NO → Building No.

This is an indirect functional dependency,
hence called Transitive functional Dependency

Properties of Functional Dependency

1) Reflexivity

if y is a subset of x then $x \to y$

2) Augmentation

if $x \to y$ then $xz \to yz$

3) Transitive

if $x \to y$ and $y \to z$ then $x \to z$

4) Union

if $x \to y$ and $x \to z$ then $x \to yz$

5) Decomposition

if $x \to yz$ then $x \to y$ and $x \to z$

6) Pseudotransitivity

if $x \to y$ and $wy \to z$ then $wx \to z$

7) Composition

if $x \to y$ and $z \to w$

then $xz \to yw$

Find Closure of Functional Dependency

* This is help to find out all the Candidate key from the table.

Eg. R (A, B, C, D)

CK = { A }

FD = { A → B, B → C, C → D }

A^+ (Closure of A) = ABCD (CK)

↓

What A Can Determined

B^+ = BCD (Not CK)

C^+ = CD (Not CK)

D^+ = D (Not CK)

A → B → C → D

B → C

C → D

if we Check $(AB)^+$ = ABCD

A → B

B → C

C → D

Prime attribute = A

Non Prime Attribute = BCD

But (AB) is not a Candidate key

Because AB is a Candidate Key is Minimal, Only A is sufficient to Determined all attribute. So AB is Super Key.

eg. $R(A, B, C, D)$ * | Ck $\{A, B, C, D\}$

$FD = \{A \to B, B \to C, C \to D, D \to A\}$

$A^+ = ABCD$ (Ck)

$B^+ = BCDA$ (Ck)

$C^+ = CDAB$ (Ck)

$D^+ = DABC$ (Ck)

Prime Attribute :- Attributes which are used in making of Candidate Key.

A, B, C, D

Non-Prime Attribute :- Remaining attributes which are unused in making of Candidate Key.

Null

eg. $R(ABCDE)$ $[Ck = \{AE, DE, BE\}]$

$FD = \{A \to B, BC \to D, E \to C, D \to A\}$

Prime attributes = A B D E

Non Prime attributes = C

* first Check those attributes which are coming in right side. Because right side attributes are determined by some attributes.

= B D C A (Determined attributes)

⇒ E (Not come in right side)

if E will come in left side then come on right side.
And for making any Candidate key E will also be determined by that attribute.

So

$E^+ = BDCAE$ (

$E^+ = EC$

Now Check

$(AE)^+ = AEBCD$ (CK)

*Now after finding first Candidate key, check either A or E present in Right Hand side

A E | $D \rightarrow A$

Replace ↓

D E

Now Check $(DE)^+ = DECAB$ (CK)

* Check if E is present in Right Hand side then Answer is No

* Now Check if D is present in Right Hand side

D E | D E | $BC \rightarrow D$

Replace ↓ | ↓

B E | C E

Now Check $(BE)^+ = BECDA$ (CK)

Now Check $(CE)^+ = CE$ (Not CK)

all attribute.

Types of Normal forms

1) 1 NF
2) 2 NF
3) 3 NF
4) BCNF (Boyce-Codd Normal Form)
5) 4NF
6) 5NF

First Normal Form (1 NF)

* A relation will be in 1 NF if it contains an atomic value.
* It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attribute.

Eg.

ID	Name	Phone
1	A	123, 345
2	B	246, 348
3	C	211

The above table is not in 1 NF because of multi-valued attribute Phone.

The Decomposition of the above table into 1 NF has been shown below -

ID	Name	Phone
1	A	123
1	A	345
2	B	246
2	B	348
3	C	211

Normalization in DBMS

* Normalization is the process of organizing the data in the database.
* Normalization is used to minimize the redundancy from a relation of set of relations. It is also used to eliminate undesirable characteristics like Insertion, Update and Deletion Anomalies.
* Normalization divides the larger table into smaller and links them using relationships.
* The normal form is used to reduce redundancy from the database table.

Data Modification anomalies can be categorized into three types -

1) **Insertion Anomaly**: It refers to when one cannot insert a new tuple into a relationship due to lack of data.
2) **Deletion Anomaly**: It refers to the situation where the deletion of data results in the unintended loss of some other important data.
3) **Updation Anomaly**: The update anomaly is when an update of a single data value requires multiple rows of data to be updated.

Second Normal Form (2 NF)

* In the 2 NF, relation must be in 1 NF.
* In the 2 NF, all non-key (Non-Prime) attributes are fully functional dependent on the primary key (CK).

Partial Dependency happens when a non-prime attribute is functionally dependent on a portion of the given candidate key.

In other words, partial dependency arises when an attribute in a table is dependent on only a portion of the primary key rather than the entire key.

2 NF

Eg. R (A B C D) — B → C, AB → D

$(AB)^+ = ABCD$

CK = AB

Prime attribute = A, B

Non Prime attribute = C, D

$B \rightarrow C$ ⇒ Partial Dependency

Because C is only Depend on B, Not on AB

Above Table is not in 2NF

Table

1) → for Candidate key, And Attribute who are depend on entire CK.

2) R_1 (A B D) — B → D

   R_2 (B, C) — B → C

} Decomposed Table

Q. R (A B C) — AB → C

$R_1(A, B)$ $R_2(B, C)$

R

a	b	c
a	1	x
b	2	y
a	3	z
c	3	z
d	3	z
e	3	z

R_1

a	b
a	1
b	2
a	3
c	3
d	3
e	3

R2

b	c
1	x
2	y
3	z

Third Normal Form (3NF)

A relation is in third Normal form, if there is no transitive dependency for non-prime attributes as well as it is in second Normal form.

* A relation will be in 3NF if it is 2NF and not contain any transitive partial dependency.

* 3NF is used to reduce the data duplication. It is also used to achieve the data integrity.

* If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form.

A relation is in 3NF if it holds atleast one of the following conditions for every non-trivial functional dependency

$X \rightarrow Y$

(Note:- This one will be helpful when Many than one Ck present)

1) X is a super key.

2) Y is a prime attribute, i.e. each element of Y is part of some candidate key.

Q. R (A B C)

FD = (A → B, B → C)

A+ = A B C (CK) (P = A, NP = B

No Partial Dependency
so it is in 2 NF.

∵ B (NP) → C (NP) (Transitive Dependency)

Transitive Dependency:- A FD from A → B is called transitive if A, B are Non Prime.

Now Decompose

R (A, B, C)

R1 (A B) R2 (B C)

Check this table is in 3NF or Not.

R (A, B, C, D)

$(AB)^+ = A\ B\ C\ D$

A, B → Prime Attribute (P)

C, D → Non Prime Attribute (NP)

There is no partial Dependency
so this table is already in 2NF.

1) (A B → C) ✓ (P → NP)

2) (C → D) → Transitive Dependency (NP → NP)

Just Because of Transitive Dependency, this table is not in 3NF.

R (A, B, C, D)

R_1 (A, B, C) — Decomposed Table — R_2 (C, D)

No Partial Dependency
No Transitive Dependency

No Partial &
No Transitive Dependency

A transitive Dependency occurs when one non prime attribute is dependent on another non prime attribute.

Eg. $Y \rightarrow Z$

Here Y and Z are Non Prime attributes

8. R (A B C D E)

FD = { A → B , C → D , B → E }

$(AC)^+$ = A C B D E (CK)

P = A , C

NP = B , D E

to check 3 NF (No Need to check Partial or transitive dependency)

for $\alpha \rightarrow \beta$

either α = super key

or β = Prime attribute

FD ① A (Not SK) → B (Not Prime) so this is not allowed in 3 NF

So this table is not into 3 NF

Now Convert this table into 3 NF

- R1 (A B E)
  - Now A → B (Key) (Non Prime)
  - B → E (Non Key) (Non Prime)
  - (Allowed)
    - R_{11} (A → B)
    - R_{12} (B → E)
- R2 (C D)
- R3 (A C)

Boyce Codd Normal form (BCNF)

* BCNF is the advance version of 3NF. It is stricter than 3NF.
* A table is in BCNF if every dependency $X \to Y$, X is the super key of the table.
* For BCNF, the table should be in 3NF, and for every FD, LHS is super key.

If we have

2NF: $\alpha \to B$ (Partial Dependency), $P \to NP$

3NF: $\alpha \to B$ (Transitive Dependency), $NP \to NP$

$\alpha \to B$ always

α : P/NP, B : NP

But if we have

$\alpha \to B$

α : Prime/Non Prime, B : Prime

It will Come into BCNF

Eg. R(A B C)

FD = {AB → C, C → B}

(AB, AC) = CK

Table is in 3NF

Because

C → B

P (Prime)

Prime attribute will be Determinant Not Determined.

Q. R (A B C)

FD: { AB → C
C → B }

$(AB)^+$ = A B C

$(AC)^+$ = A B C

Now (AB, AC) = CK

Now Check for BCNF

1) AB → C (Allow)
(Super key)

2) C → B (Not Allow)
(Not a Super key)

So it is Not into BCNF

Decomposition into BCNF

(Compulsory) R1 (C B)
CK R2 (A C)
CK R3 (A B)

Both table have Common attribute (C) and C is Key attribute in R1

Both table have Common attribute (B) But B is not a Key attribute.

So for Decomposition we will allow R1 and R2