

111121

UNIT-III

SQL

1. Find the name and age of all sailors

Select .DISTINCT (To we only one time in relation)

SELECT → from select list

FROM → from from list

Qualification → from where classe

select distinct name, age

from sailors

2. Find all sailors with rating above 7.

Select \* from sailors

where rating > 7

3. Find the name of sailors who have reserved boat 103.

Select s.sname

from sailors s, Reserve R

where S.sid = R.sid AND R.bid = 103

4. Find the name of sailors who have reserved red boat.

Select s.sname

from sailors s, Reserve R, Boat B

where B.colour = Red AND B.bid = R.bid

AND s.sid = R.sid

5. Find the colour of boats reserved by lubber.

Select B.colour

From Sailor S, Reserve R, Boat B

where S.sname = lubber AND S.sid = R.sid

AND R.bid = B.bid.

Date = 25/11/98

6. Find the name of sailors who have reserved atleast one boat.

Select s.sname

From Sailor S, Reserve R

where S.sid = R.sid

25/11/21

Expression and strings:

Complete increments for the rating of person who have sailed two different boats on the same day.

Select rating + 1 as rating, name

From Sailors, Reserve

where S.sid = R.sid AND

Date = 9/8/98

Each item in a select list can be a form of expression as column name, where expression is arithmetic or string expression over the column names.

→ The string expression we can use comparison operators.

→ If we need to sort strings by order other than alphabets.

Like operator: SQL supports pattern matching through the like operator along with the use of wild card.

(or) % that symbol represent 0 or more characters or arbitrary characters and '-' (hyphen) symbol for exactly one arbitrary character.

'like a=%'

'like 'a%'

It finds all like 'a%'.

It finds all like '%a'.

→ Like 'a%' find any values that starts with a character of string.

→ Like '%a' find any value that will be last character of a string.

→ Like '%-%' its means to find the second position of character.

→ Like '!a-%' find two characters of string

→ 'a--%' three characters of string

29/11/21

Find the name of sailors who have reserved red or green boat.

From sailors s, reserve R, Boat B  
where s.sid = R.sid AND R.bid = B.bid  
AND colour = "Red" UNION  
AND colour = "Green"

From sailors s, reserve R, Boat B  
where s.sid = R.sid AND R.bid = B.bid  
AND colour = "Red" OR  
AND colour = "Green"

Find the name of sailors who have reserved red and green boat

From sailors s, reserve R, Boat B  
where s.sid = R.sid AND R.bid = B.bid  
AND colour = "Red"  
AND colour = "Green" INTERSECTION

From sailors s, reserve R, Boat B  
where s.sid = R.sid AND R.bid = B.bid  
AND colour = "Red"  
AND colour = "Green"

Find the sid of all sailors who have reserved red colour boat but not in green colour boat

Select sname, sid

From Sailors S, Reserve R, Boat B

where s.sid = R.sid AND R.bid = B.bid  
AND colour "Red"

EXCEPT

Select s.sid

From Sailors S, Reserve R, Boat B

where s.sid = R.sid AND B.bid = B.bid  
AND colour = "Green"

### Nested query's / queries:

A nested query is the query that has another query embedded within it. The sub

query is applied in the where class

Ex: Find the name of sailors who have

reserved boat 103.

Nested: select s.sname

From Sailors

where s.sid IN (select R.sid  
From reserves  
where R.bid = 103)

General: select s.sname

From Sailors, reserve

where s.sid = R.sid AND R.bid = 103

Find the name of sailors who have not reserved a red boat, does not have a bid.

Select s.sname  
From sailors  
where s.sid NOT IN [ select R.sid  
From Reserves, Boats  
where B.colour = 'red'  
R.bid = B.bid ]

30/11/21

Correlated nested query:

The nested query should be independent

Find the name of sailors who have reserved boat number 103.

select \* from reserves  
where R.bid = 103

select sname  
From sailors  
where s.sid IN ( select \* from reserves  
where R.bid = 103 )

Set comparison operators:

The SQL supports  $\in$ ,  $\exists$ ,  $\forall$ ,  $\neq$ ,  $\geq$ ,  $\leq$ .

ANY AND ALL Operations

Find the sailors whose rating is better than some sailors called horatio

SELECT sname FROM sailors WHERE rating > (SELECT rating FROM sailors WHERE sname = 'horatio')

select s.sid  
from sailors  
where s.rating > any ( select rating  
from sailors  
where s.name = haralda)

Find the sailors with the highest rating.

select s.sid  
from sailors  
where s.rating > all ( select rating  
from sailors)

11/2/21

Aggregate operators :- (in select class) bag by group

1. COUNT

total values (both bag) function

2. SUM

sum of group - function

3. AVG

average of group - function

4. MIN

min group - function

5. MAX

max group - function

(DISTINCT)  $\rightarrow$  A represents field of relation

Find the average age of all sailors.

select Avg(s.age)

level position

from sailors

Find the average age of sailors with rating 10

select Avg(s.age)

filter stat

from sailor s

rating

where s.rating = 10

Find the name and age of oldest sailor.

select MAX (age) s.name  
from sailor s

(oldest - common abjects)

Count the no. of sailors, all records soft table

select count (\*)  
from sailors s

Count the no. of different sailors name

select count [(DISTINCT)s.name]  
from sailors s

21/2/21

Group by and having clause

select [Distinct] select list

From from clause

qualification - where class

group of rows - group by

where [qualification - having clause]

Find the age of youngest sailors for each boat

rating level.

(Q2-Q3) pA to be

select MIN[s.age]

From sailors to Q2 Q3 Q4 Q5 Q6 Q7 Q8 Q9 Q10 Q11 Q12 Q13 Q14 Q15 Q16 Q17 Q18 Q19 Q20 Q21 Q22 Q23 Q24 Q25 Q26 Q27 Q28 Q29 Q30 Q31 Q32 Q33 Q34 Q35 Q36 Q37 Q38 Q39 Q40 Q41 Q42 Q43 Q44 Q45 Q46 Q47 Q48 Q49 Q50 Q51 Q52 Q53 Q54 Q55 Q56 Q57 Q58 Q59 Q60 Q61 Q62 Q63 Q64 Q65 Q66 Q67 Q68 Q69 Q70 Q71 Q72 Q73 Q74 Q75 Q76 Q77 Q78 Q79 Q80 Q81 Q82 Q83 Q84 Q85 Q86 Q87 Q88 Q89 Q90 Q91 Q92 Q93 Q94 Q95 Q96 Q97 Q98 Q99 Q100 Q101 Q102 Q103 Q104 Q105 Q106 Q107 Q108 Q109 Q110 Q111 Q112 Q113 Q114 Q115 Q116 Q117 Q118 Q119 Q120 Q121 Q122 Q123 Q124 Q125 Q126 Q127 Q128 Q129 Q130 Q131 Q132 Q133 Q134 Q135 Q136 Q137 Q138 Q139 Q140 Q141 Q142 Q143 Q144 Q145 Q146 Q147 Q148 Q149 Q150 Q151 Q152 Q153 Q154 Q155 Q156 Q157 Q158 Q159 Q160 Q161 Q162 Q163 Q164 Q165 Q166 Q167 Q168 Q169 Q170 Q171 Q172 Q173 Q174 Q175 Q176 Q177 Q178 Q179 Q180 Q181 Q182 Q183 Q184 Q185 Q186 Q187 Q188 Q189 Q190 Q191 Q192 Q193 Q194 Q195 Q196 Q197 Q198 Q199 Q199 Q200 Q201 Q202 Q203 Q204 Q205 Q206 Q207 Q208 Q209 Q210 Q211 Q212 Q213 Q214 Q215 Q216 Q217 Q218 Q219 Q220 Q221 Q222 Q223 Q224 Q225 Q226 Q227 Q228 Q229 Q229 Q230 Q231 Q232 Q233 Q234 Q235 Q236 Q237 Q238 Q239 Q239 Q240 Q241 Q242 Q243 Q244 Q245 Q246 Q247 Q248 Q249 Q249 Q250 Q251 Q252 Q253 Q254 Q255 Q256 Q257 Q258 Q259 Q259 Q260 Q261 Q262 Q263 Q264 Q265 Q266 Q267 Q268 Q269 Q269 Q270 Q271 Q272 Q273 Q274 Q275 Q276 Q277 Q278 Q279 Q279 Q280 Q281 Q282 Q283 Q284 Q285 Q286 Q287 Q288 Q289 Q289 Q290 Q291 Q292 Q293 Q294 Q295 Q296 Q297 Q298 Q299 Q299 Q300 Q301 Q302 Q303 Q304 Q305 Q306 Q307 Q308 Q309 Q309 Q310 Q311 Q312 Q313 Q314 Q315 Q316 Q317 Q318 Q319 Q319 Q320 Q321 Q322 Q323 Q324 Q325 Q326 Q327 Q328 Q329 Q329 Q330 Q331 Q332 Q333 Q334 Q335 Q336 Q337 Q338 Q339 Q339 Q340 Q341 Q342 Q343 Q344 Q345 Q346 Q347 Q348 Q349 Q349 Q350 Q351 Q352 Q353 Q354 Q355 Q356 Q357 Q358 Q359 Q359 Q360 Q361 Q362 Q363 Q364 Q365 Q366 Q367 Q368 Q369 Q369 Q370 Q371 Q372 Q373 Q374 Q375 Q376 Q377 Q378 Q379 Q379 Q380 Q381 Q382 Q383 Q384 Q385 Q386 Q387 Q388 Q389 Q389 Q390 Q391 Q392 Q393 Q394 Q395 Q396 Q397 Q398 Q398 Q399 Q399 Q400 Q401 Q402 Q403 Q404 Q405 Q406 Q407 Q408 Q409 Q409 Q410 Q411 Q412 Q413 Q414 Q415 Q416 Q417 Q418 Q419 Q419 Q420 Q421 Q422 Q423 Q424 Q425 Q426 Q427 Q428 Q429 Q429 Q430 Q431 Q432 Q433 Q434 Q435 Q436 Q437 Q438 Q439 Q439 Q440 Q441 Q442 Q443 Q444 Q445 Q446 Q447 Q448 Q449 Q449 Q450 Q451 Q452 Q453 Q454 Q455 Q456 Q457 Q458 Q459 Q459 Q460 Q461 Q462 Q463 Q464 Q465 Q466 Q467 Q468 Q469 Q469 Q470 Q471 Q472 Q473 Q474 Q475 Q476 Q477 Q478 Q479 Q479 Q480 Q481 Q482 Q483 Q484 Q485 Q486 Q487 Q488 Q489 Q489 Q490 Q491 Q492 Q493 Q494 Q495 Q496 Q497 Q498 Q498 Q499 Q499 Q500 Q501 Q502 Q503 Q504 Q505 Q506 Q507 Q508 Q509 Q509 Q510 Q511 Q512 Q513 Q514 Q515 Q516 Q517 Q518 Q519 Q519 Q520 Q521 Q522 Q523 Q524 Q525 Q526 Q527 Q528 Q529 Q529 Q530 Q531 Q532 Q533 Q534 Q535 Q536 Q537 Q538 Q539 Q539 Q540 Q541 Q542 Q543 Q544 Q545 Q546 Q547 Q548 Q549 Q549 Q550 Q551 Q552 Q553 Q554 Q555 Q556 Q557 Q558 Q559 Q559 Q560 Q561 Q562 Q563 Q564 Q565 Q566 Q567 Q568 Q569 Q569 Q570 Q571 Q572 Q573 Q574 Q575 Q576 Q577 Q578 Q579 Q579 Q580 Q581 Q582 Q583 Q584 Q585 Q586 Q587 Q588 Q589 Q589 Q590 Q591 Q592 Q593 Q594 Q595 Q596 Q597 Q598 Q598 Q599 Q599 Q600 Q500 Q501 Q502 Q503 Q504 Q505 Q506 Q507 Q508 Q509 Q509 Q510 Q511 Q512 Q513 Q514 Q515 Q516 Q517 Q518 Q519 Q519 Q520 Q521 Q522 Q523 Q524 Q525 Q526 Q527 Q528 Q529 Q529 Q530 Q531 Q532 Q533 Q534 Q535 Q536 Q537 Q538 Q539 Q539 Q540 Q541 Q542 Q543 Q544 Q545 Q546 Q547 Q548 Q549 Q549 Q550 Q551 Q552 Q553 Q554 Q555 Q556 Q557 Q558 Q559 Q559 Q560 Q561 Q562 Q563 Q564 Q565 Q566 Q567 Q568 Q569 Q569 Q570 Q571 Q572 Q573 Q574 Q575 Q576 Q577 Q578 Q579 Q579 Q580 Q581 Q582 Q583 Q584 Q585 Q586 Q587 Q588 Q589 Q589 Q590 Q591 Q592 Q593 Q594 Q595 Q596 Q597 Q598 Q598 Q599 Q599 Q600 Q601 Q602 Q603 Q604 Q605 Q606 Q607 Q608 Q609 Q609 Q610 Q611 Q612 Q613 Q614 Q615 Q616 Q617 Q618 Q619 Q619 Q620 Q621 Q622 Q623 Q624 Q625 Q626 Q627 Q628 Q629 Q629 Q630 Q631 Q632 Q633 Q634 Q635 Q636 Q637 Q638 Q639 Q639 Q640 Q641 Q642 Q643 Q644 Q645 Q646 Q647 Q648 Q649 Q649 Q650 Q651 Q652 Q653 Q654 Q655 Q656 Q657 Q658 Q659 Q659 Q660 Q661 Q662 Q663 Q664 Q665 Q666 Q667 Q668 Q669 Q669 Q670 Q671 Q672 Q673 Q674 Q675 Q676 Q677 Q678 Q679 Q679 Q680 Q681 Q682 Q683 Q684 Q685 Q686 Q687 Q688 Q689 Q689 Q690 Q691 Q692 Q693 Q694 Q695 Q696 Q697 Q698 Q698 Q699 Q699 Q700 Q701 Q702 Q703 Q704 Q705 Q706 Q707 Q708 Q709 Q709 Q710 Q711 Q712 Q713 Q714 Q715 Q716 Q717 Q718 Q719 Q719 Q720 Q721 Q722 Q723 Q724 Q725 Q726 Q727 Q728 Q729 Q729 Q730 Q731 Q732 Q733 Q734 Q735 Q736 Q737 Q738 Q739 Q739 Q740 Q741 Q742 Q743 Q744 Q745 Q746 Q747 Q748 Q749 Q749 Q750 Q751 Q752 Q753 Q754 Q755 Q756 Q757 Q758 Q759 Q759 Q760 Q761 Q762 Q763 Q764 Q765 Q766 Q767 Q768 Q769 Q769 Q770 Q771 Q772 Q773 Q774 Q775 Q776 Q777 Q778 Q779 Q779 Q780 Q781 Q782 Q783 Q784 Q785 Q786 Q787 Q788 Q789 Q789 Q790 Q791 Q792 Q793 Q794 Q795 Q796 Q797 Q798 Q798 Q799 Q799 Q800 Q801 Q802 Q803 Q804 Q805 Q806 Q807 Q808 Q809 Q809 Q810 Q811 Q812 Q813 Q814 Q815 Q816 Q817 Q818 Q819 Q819 Q820 Q821 Q822 Q823 Q824 Q825 Q826 Q827 Q828 Q829 Q829 Q830 Q831 Q832 Q833 Q834 Q835 Q836 Q837 Q838 Q839 Q839 Q840 Q841 Q842 Q843 Q844 Q845 Q846 Q847 Q848 Q849 Q849 Q850 Q851 Q852 Q853 Q854 Q855 Q856 Q857 Q858 Q859 Q859 Q860 Q861 Q862 Q863 Q864 Q865 Q866 Q867 Q868 Q869 Q869 Q870 Q871 Q872 Q873 Q874 Q875 Q876 Q877 Q878 Q879 Q879 Q880 Q881 Q882 Q883 Q884 Q885 Q886 Q887 Q888 Q889 Q889 Q890 Q891 Q892 Q893 Q894 Q895 Q896 Q897 Q898 Q898 Q899 Q899 Q900 Q901 Q902 Q903 Q904 Q905 Q906 Q907 Q908 Q909 Q909 Q910 Q911 Q912 Q913 Q914 Q915 Q916 Q917 Q918 Q919 Q919 Q920 Q921 Q922 Q923 Q924 Q925 Q926 Q927 Q928 Q929 Q929 Q930 Q931 Q932 Q933 Q934 Q935 Q936 Q937 Q938 Q939 Q939 Q940 Q941 Q942 Q943 Q944 Q945 Q946 Q947 Q948 Q949 Q949 Q950 Q951 Q952 Q953 Q954 Q955 Q956 Q957 Q958 Q959 Q959 Q960 Q961 Q962 Q963 Q964 Q965 Q966 Q967 Q968 Q969 Q969 Q970 Q971 Q972 Q973 Q974 Q975 Q976 Q977 Q978 Q979 Q979 Q980 Q981 Q982 Q983 Q984 Q985 Q986 Q987 Q988 Q988 Q989 Q989 Q990 Q991 Q992 Q993 Q994 Q995 Q996 Q997 Q998

- The select list in the select class consists of a list of column names and list of terms the form aggregate operator of column name as new name
- we already saw as used to rename the output columns.
- every column that appears in group list, the reason is that each row in the result of the query corresponding to one group.
- The expression appearing in the group qualifications is a having clause and they must have a single value for group.

For example:

Find the age of youngest sailor whose is eligible to vote and for each rating level it atleast took two such sailors.

Select s.rating; min(s.age) As Newage

From Sailors

where s.age >= 18

group by s.rating

having count(\*) > 1;

To column in sailors does specify which column

should be selected. So, we can use group by

statement to group the rows which have same

attribute values. So, we can use group by

## Procedural language / structural query language

4/12/21

- declare (declaration statement) variables
- begin (executable statements)
- exceptions (exceptional handling) - run-time errors
- end

→ program will be executed with PL/SQL

SQL, PL/SQL or Oracle block creates view

### Example:

Addition of two numbers

SQL > Add two numbers

SQL > Declare

    variable declared in the block

    a number(10);

    b number(10);

    c number(10);

    if local variable does not have size of 3 digits

SQL > Begin

    c = a+b;

    dbms\_output.put\_value ("sum of values : "||c);

end;

/

SQL > execute block

→ PL/SQL is basically procedural language which provides functionality of decision making.

→ The PL/SQL can execute a number of queries in one block using single command.

→ One can create PL/SQL unit such as procedures, functions, triggers, cursors, which stored in database

for reuse by applications.

→ PL/SQL made up of 4 blocks.

Trigger in active database: (automatically run when an event occurs in database server)

→ Triggers are a program which is automatically executed when some events occurred.

→ A trigger can be defined as program that is executed by dbms.

→ Trigger is like an event which occurs whenever a change is done to the tables or columns of the table.

→ Triggers are written to be executed in DDL, DML and database operations.

General forms of triggers:

1. Event

2. Condition

3. Action

→ THE EVENT it describes the modification terms on DB which lead to the activation of the trigger. The events are DDL, DML and data operation.

→ condition:- This are used to specify the actions to be taken when the corresponding event occurs. The condition evaluates true. The respect action to be taken otherwise rejected.

→ ACTION specifies when the corresponding event occurs and condition evaluates to true.

- An 'action' is collection of SQL statements that are executed as part of trigger.

Syntax structure of trigger:

create or update trigger <triggername>

before or after of <table> on <table name>

insert or update or delete <table name>

For each row which fires on each of update,

which <condition for trigger to get execute>

Declare

<declaration part> str construct and applicable

Begin

<execution part> expect to except handle

Exception

<exception part>

End;

7/12/21

## Triggers

Following program creates a row level trigger for the customer table that would be performed for - insert, update, delete operations. Performed on the customer table. This trigger will display the salary differences between old & new salary statements.

- create or replace trigger salary

'before' on 'customer' for each row

when(new.ID > 0)

Declare

sal-diff number;

begin

sal-diff := new.Sal - old.Sal;

dbms-output.put-values("old sal:", 11: old.Sal);  
dbms-output.put-values("new sal:", 11: new.Sal);  
dbms-output.put-values("new sal:", 11: sal-diff);

end;

/

update the table. salary incremented by 1000Rs.

create or replace trigger salary

before update on customer

for each row

if

if

if

• Declare

beginning

level Total - rows := number of rows updated  
of bloks total size remains soft for update  
begin  
update customer;  
set salary = salary + 1000;  
user is blo account password  
If SQL% not found them  
dbms-output.put-values ("not updated");  
elseif SQL% found  
total-rows = SQL% row count;  
dbms-output.put-values ("updated");  
endif;  
end;

/

8/12/21

- Scheme refinement & Normal forms: f1q, f2q, f3q, f4q, f5q, f6q
1. Redundancy (multivalued attr)
  2. Update anomalies (problem)
  3. Insert anomalies
  4. Delete anomalies

Employee: pt balance initial value deficit soft str by

eid	ename	salary	age	hourly-wages	hours-worked
111	AA	25,000	8	10	40
222	BB	30,000	8	10	30
333	CC	35,000	5	7	35
444	DD	40,000	5	7	42
555	EE	45,000	8	10	32

$\text{Emp}(\text{eid}, \text{ename}, \text{salary}, \text{age}, \text{hoursworked})$

$\text{wages}(\text{age}, \text{hourly-wages})$

age	hourly-wages
8	10
5	7

Eid	Ename	salary	age	hours worked
111	AA	25,000	8	40
222	BB	30,000	8	30
333	CC	35,000	5	35
444	DD	40,000	5	42
555	EE	45,000	8	32

### Schema Refinement:

The main cause of schema refinement

eliminate the redundant storage by the decompose relation.

- The problem caused by redundancy.

- Redundancy is a method of storing the same information repeatedly. It means storing the same data more than 1 place within the database and it can lead several problems then the problems are

1. Redundancy

2. Update anomalies

3. Insert anomalies

4. Delete anomalies

## 1. Redundancy or Redundancy storage

It removes the multi-value attributes, It means some tuples or information (values) is stored repeatedly.

## 2. Update anomalies:-

Suppose if we update 1 row then the dbms will update more than 1 rows.

## 3) Insertion Anomalies:-

When allows insertion for already exist records again inserted.

## 4) Deletion Anomalies:-

When more than one record is deleted instead of specified one.

- overcome these problems using decomposition.

- The decomposition is a relation schema  $R$  can be decomposed into a collection of smaller relations ( $R_1, R_2, R_3, \dots, R_n$ ) to eliminate redundant information.

Increase  $|R_i| \leq |R|$  for all  $i$ .

Thus each tuple of  $R$  is stored in different tables.

Table  $R_1$  stores attribute  $A_1$ .

Table  $R_2$  stores attribute  $A_2$ .

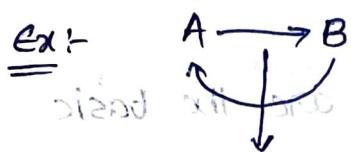
Table  $R_3$  stores attribute  $A_3$ .

Table  $R_4$  stores attribute  $A_4$ .

9/12/21

## Functional dependency:

The functional dependency is a relationship that exists between 2 attributes. The 2 attributes are prime key, non key attribute in the table.



self dependency  $\rightarrow$   $(\rightarrow) = \text{determine}$   
dependency symbol

symbol determine si fi kia slot

(or)

dependency

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

known about Non-key attribute or prime key

Functional dependency  $\rightarrow$  2 types

1. Trivial functional dependencies

2. Non-trivial

If  $A \rightarrow B$  has trivial FID then  $Y \subseteq X$

If  $\{B_1, B_2\}$  is a subset of  $A$   $= Y \subseteq X$

Ex:-  $\{ \text{emp Id}, \text{emp name} \}$  is a subset of  $A$   $= Y \subseteq X$   
 $\text{emp name}$  determines emp Id.

$X \subseteq Y$  and  $Y \subseteq X$

$X \subseteq Y$  and  $Y \subseteq X$   $\Rightarrow X = Y$

$A \rightarrow B$  has non trivial FD

If  $B$  is not a subset of  $A$  ~~subset of A~~ ~~logical~~

Ex:  $\{Address, Name\} \rightarrow DOB$  ~~subset of~~  $Name \rightarrow DOB$

FD Inference rule (IR) has been used to prove

→ Armstrong Axioms

(axioms of IR)

The Armstrong axioms are the basic

postulates

inference role and it is used to conclude

(so)

functional dependency on the relational database.

→ The inference rule is a type of assertion  
and it can be applied to set of functional dependency to derive other functional dependency

→ In functional dependencies there are 6 types of IR's

### 1. Reflexive Rule (IR<sub>1</sub>)

If  $X \supseteq Y$  then  $X \rightarrow Y$  and  $g \leftarrow A$

Ex:  $X = \{A, B, C, D, E\}$ ;  $Y = \{A, B, C\}$

### 2. Augmentation Rule (IR<sub>2</sub>)

(partial dependency)

• BI  $\leftarrow$  no Z

If  $X \rightarrow Y$  then  $XZ \rightarrow YZ$

R(ABCD), If  $A \rightarrow B$  then  $AC \rightarrow BC$

### 3) Transitive rule (IR<sub>3</sub>)

$x \rightarrow Y$  and  $Y \rightarrow Z$  then  $x \rightarrow Z$

CLPPEP

### 4) Union rule (IR<sub>4</sub>)

$x \rightarrow Y$  and  $x \rightarrow Z$  then  $x \rightarrow YZ$

### 5) Decomposition rule (IR<sub>5</sub>)

$x \rightarrow YZ$  then  $x \rightarrow Y$ , and  $x \rightarrow Z$

### 6) pseudo transitive rule (IR<sub>6</sub>)

If  $x \rightarrow Y$  and  $YZ \rightarrow W$  then  $x \rightarrow W$

1312121

### Normalization:

- organizing the data
- Reduce the redundancy
- eliminate the undesirable characteristics
- large tables to small table.

### 1st normal form:

- Automatic values (not multiple values, single value only).
- cannot use multivalue attribute
- Must use single value attribute in the value

- disallow multivalue attribute & composite key.

### 2nd normal form:

candidate key didn't get decomposed

SID	Sname	SPhone.no.	Sage	Deparment
1	A	1234567	10	(FST) auto mobile
2	B	11121314	20	(CSE) computer
3	C	15161718 19202122	30	(ECE) electronic

After normal applying normal values (Atomic values):

		(SCE) stud. activities	Deparment
1	A	1234567	10
1	A	78910	10
2	B	11121314	20
3	C	15161718	30
3	C	19202122	30

- 2nd Normal form
- All non-key attributes are fully functional dependency of primary key.
- 2NF is in 1st NF

Subject	Teacher id	subject	Age	see table
20	20	A	30	Teacher
20		B	30	J. ID → subject
30		C	35	Teacher
40		D	34	Decomposition
40		E	38	

- Decompose the table or relation.

T.Id	Age	T.Id	subject
20	30	20	A
30	35	20	B
40	38	30	C
40	36	40	D
		40	E

### Composite key:

Composite key is a primary key composed of multiple columns, used to identify record uniquely.

3<sup>rd</sup> NF:-

14/12/21

1 3NF, it is in 2NF

$x \rightarrow y$      $y \rightarrow z$      $x \rightarrow z$

2. Not contain any transitive dependency

3 Reduce the data duplication

↑ non-prime attributes

3NF & 4NF				
Eid	ename	e.zipcode	e.city	e.state
1	A	101	AP	AM
2	B	102	UP	Noida
3	C	103	UP	Noida
4	D	104	MP	Bhopal
5	E	105	TS	Hyd

conditions:-

superkey  $\rightarrow \{eid\}, \{eid, ename\}$

1) X is a superkey

$\{eid, ename, ezip\} \dots \dots$

2) Y is a prime key

$c.k \rightarrow \{eid\}$

4. A relation is in 3rd NF if it holds at least one of the following condition for non-trivial functional dependency of  $X \rightarrow Y$

i) X is a super key

ii) Y is a prime key (part of candidate key)

iii) X is a superset of Super key

eid	e.name	e.zipcode
1	A	101
2	B	102
3	C	103
4	D	104
5	E	105

e.zip	e.city	e.state
101	Mumbai	AM
102	UP	Noida
103	UP	Noida
104	MP	Bhopal
105	TS	Hyd

$\{e.id, e.name\}$  &  $\{e.zipcode\}$  are superkeys.  $\rightarrow$   $\{e.zip\}$ ,  $\{e.city\}$ ,  $\{e.state\}$  are non-prime attributes

$\{e.id, e.zip\} \rightarrow$  candidate key. (prime key)

$\{e.zip\} \rightarrow \{e.city, e.state\}$

$\{e.id\} \rightarrow \{e.zip\}$

$\{e.id\} \rightarrow \{e.city, e.state\}$

Non-prime attributes are functionally dependent on prime key

15/12/21

equivalence of BCNF

BCNF: If  $x \rightarrow y \rightarrow z$  -  $x$  is superkey and  $y \rightarrow z$

- BCNF - boyce codd normal form is stricter than NF

- every FD is  $x \rightarrow y$ ,  $x$  is always superkey.

Ex: There is a company where employees works more than one department.

Eid	e.country	e.dept	Dept-type	emp-dept-no
200	In	Design	D3	20
200	In	test	D3	31
300	UK	store	D2	30
300	UK	Developing	D2	31

super key:  $Eid \rightarrow e\cdot country$

$e\cdot dept \rightarrow \{Dept-type, Emp-dept-no\}$

candidate key:  $\{Eid, E.dept\}$

e.dept	Dept-type	emp-dept-no
Design	D3	20
Test	D3	21
store	D2	30
developing	D2	31

Eid	e.dept
200	Design
200	test
300	store
300	developing

$\{e\cdot dept\}$

Eid	e-C
200	In
300	UK

$\{Eid\}$

$\{Eid, e\cdot dept\}$

Eid	e-C
200	In
300	UK

## Properties of decomposition:

- lossless join decomposition -  $R \Rightarrow R_1 \& R_2 \Rightarrow R_1 \bowtie R_2 \Rightarrow R$

- dependency preserving decomposition

$$R(ABCD) \rightarrow R_1(ABC) \& R_2(DD)$$

cause dependency  $A \rightarrow BC$  due to  $A \rightarrow D$  has to be present in  $R_1$

$$A \rightarrow B$$

$$A \rightarrow C$$

1612121

4<sup>th</sup> normal form:

- the relation will be in BCNF

- cannot allow multiple value dependency (MVD)

stu-id	subject	Hobby	$A \rightarrow B$
21	com	Dance	$21 \rightarrow 1$
21	maths	sing	$21 \rightarrow 2,3$
34	chem	dance	23 is depended on PK
74	Bio	cricket	where 21 23 are
59	phy	Hockey	independent of each other

stu-id	subject
21	com
21	maths
34	chem
74	Bio
59	phy

stu-id	hobby
21	Dance
21	sing
34	dance
74	cricket
59	Hockey

5<sup>th</sup> normal form :- (or) project join NF (PJNF)

- A relation is in 4<sup>th</sup> NF

- not contain any join dependency

- Join should be a lossless join

To avoid redundancy to achieve no deletion

subject	lecture	semister
com	A	1
com	B	1
math	B	1
math	C	2
chem	D	1

Decompose the table  $R \rightarrow R_1, R_2 \& R_3$

$R_1 = \{ \text{sem}, \text{subject} \}$

$R_2 = \{ \text{sub}, \text{lecture} \}$

$R_3 = \{ \text{sem}, \text{lecture} \}$

$R \rightarrow R, R_1 \& R_2 \rightarrow R$

sub	lecture
com	A
com	B
math	B
math	C
chem	D

sem	subject	sem	lecture
1	com	1	A
1	math	1	B
2	math	2	C
	chem	D	D

obtained - using - obtained - X

(Y) formed - needs