

Database Systems – I

BIT Semester 2

3

DML - Data Manipulation Language

Working with database using DML

1. Database System Environment (5MCQs)
2. Integrity Constraints and DDL (5MCQs)
3. Working with database using DML (10MCQs)
Select, Insert, Update, Delete
4. Relation Algebra (6MCQs)
5. Database Designing Process with ER Diagrams (8MCQs)
6. Normalization (5MCQs)
7. Views and Security with DCL (4MCQs)
8. Execute duties of a Database Administrator

Base Plan

- | | Class | Date |
|--|-------|------------|
| 1. <input type="checkbox"/> Select | [1] | [1/8/19] |
| <input type="checkbox"/> Column Selecting (Projection) - Alias | | |
| <input type="checkbox"/> Row Selecting (Selection) – SQL Operators : Arithmetic/Comparison/Logical | | |
| <input type="checkbox"/> Nested Queries (Sub Queries) | | |
| <input type="checkbox"/> 2014-07 <input type="checkbox"/> 2014-28 <input type="checkbox"/> 2014-24 <input type="checkbox"/> 2014-25 <input type="checkbox"/> 2011-20 <input type="checkbox"/> 2011-21 | | |
| <input type="checkbox"/> 2011-32 <input type="checkbox"/> 2017-31 <input type="checkbox"/> 2016-19 <input type="checkbox"/> 2015-08 | | |
| <input type="checkbox"/> DISTINCT Keyword | | |
| <input type="checkbox"/> 2011-30 | | |
| <input type="checkbox"/> Ordering (Sorting) – Ascending/Descending | | |
| <input type="checkbox"/> 2010 - 40 <input type="checkbox"/> 2014-30 <input type="checkbox"/> 2018-23 <input type="checkbox"/> 2017-25 | | |
| <input type="checkbox"/> Database Functions – Aggregate (Group functions)/Scalar Functions | | |
| <input type="checkbox"/> 2013-18 <input type="checkbox"/> 2018-17 <input type="checkbox"/> 2017-33 | | |
| <input type="checkbox"/> 2016-20 <input type="checkbox"/> 2017-14 | | |
| <input type="checkbox"/> Grouping – HAVING Keyword/Priority Order | | |
| <input type="checkbox"/> 2014 – 23 | | |
| <input type="checkbox"/> 2015-09 <input type="checkbox"/> 2015-10 <input type="checkbox"/> 2012-32 <input type="checkbox"/> 2011-29 <input type="checkbox"/> 2016-16 <input type="checkbox"/> 2016-18 | | |
| <input type="checkbox"/> 2016-21 <input type="checkbox"/> 2016-22 <input type="checkbox"/> 2015-16 | | |
| <input type="checkbox"/> Wild Card Characters | | |
| <input type="checkbox"/> 2011-28 | | |
| <input type="checkbox"/> LIMIT and TOP Keyword | | |
| <input type="checkbox"/> Join | | |
| <input type="checkbox"/> Cross join | | |
| <input type="checkbox"/> 2015-18 <input type="checkbox"/> 2014-19 <input type="checkbox"/> 2015-09 <input type="checkbox"/> 2015-16 | | |
| <input type="checkbox"/> Inner join | | |
| <input type="checkbox"/> 2010-28 <input type="checkbox"/> 2014-20 <input type="checkbox"/> 2017-31 <input type="checkbox"/> 2013-23 <input type="checkbox"/> 2011-26 <input type="checkbox"/> 2015-14 <input type="checkbox"/> 2016-16 | | |
| <input type="checkbox"/> 2016-18 <input type="checkbox"/> 2010-30 <input type="checkbox"/> 2011-20 <input type="checkbox"/> 2011-21 <input type="checkbox"/> 2013-24 <input type="checkbox"/> 2012-24 | | |
| <input type="checkbox"/> 2012-25 <input type="checkbox"/> 2013-21 <input type="checkbox"/> 2014-29 | | |
| <input type="checkbox"/> Natural Join | | |
| <input type="checkbox"/> 2018-28 <input type="checkbox"/> 2017-32 | | |
| <input type="checkbox"/> Outer join | | |
| <input type="checkbox"/> Left Outer Join | | |
| <input type="checkbox"/> 2018-39 <input type="checkbox"/> 2015-08 <input type="checkbox"/> 2013-20 | | |
| <input type="checkbox"/> Right Outer Join | | |
| <input type="checkbox"/> 2017-44 <input type="checkbox"/> 2010-23 <input type="checkbox"/> 2014-22 <input type="checkbox"/> 2016-16 <input type="checkbox"/> 2013-19 | | |
| 2. <input type="checkbox"/> Insert (Single, Multiple) | [] | [] |
| <input type="checkbox"/> 2015-07 <input type="checkbox"/> 2014-21 <input type="checkbox"/> 2011-27 <input type="checkbox"/> 2011-31 <input type="checkbox"/> 2011-40 | | |
| <input type="checkbox"/> 2010-19 <input type="checkbox"/> 2010-20 <input type="checkbox"/> 2009-25 <input type="checkbox"/> 2007-24 <input type="checkbox"/> 2006-23 | | |
| 3. <input type="checkbox"/> Update (All rows, Selected Rows) | [] | [] |
| <input type="checkbox"/> 2017-27 <input type="checkbox"/> 2013-39 <input type="checkbox"/> 2012-26 <input type="checkbox"/> 2012 -28 <input type="checkbox"/> 2009 -27 | | |
| <input type="checkbox"/> 2007-21 <input type="checkbox"/> 2006-31 <input type="checkbox"/> 2005-31 | | |
| 4. <input type="checkbox"/> Delete (All rows, Selected Rows) | [] | [] |
| <input type="checkbox"/> 2016-25 <input type="checkbox"/> 2013 – 25 <input type="checkbox"/> 2011 – 22 <input type="checkbox"/> 2010-24 <input type="checkbox"/> 2009-29 | | |
| <input type="checkbox"/> 2008 – 10 | | |

Select Statement

Consider following Table when execute the queries.

Student					
stuNo	stuName	gender	age	tpNo	city
1	Namal	male	22	0013456702	Gampaha
2	Kumara	male	27	0217896465	Boralla
3	Vishaka	female	25	0799956662	Boralla
4	Anuradha	male	23	0567856705	Nawala
5	Nirmala	female	18	0987034702	Gampaha
6	Amal	male	32	0348787890	Maharagama

Employee

id	firstname	lastname	incentiverate	basicsalary	vehicleallownce
1	Nimal	Perera	10	20000.00	2500.00
2	Sunil	Silva	7.5	18000.00	2000.00
3	Nimali	Sirisena	12.5	15000.00	1500.00
4	Ruwan	Gamage	20	35000.00	15000.00

Visitors

firstName	lastName
Nimal	Wimalachandra
Sunil	Basnayake
Suren	Wikramarachchi
Namal	Manampitiya
Dulanga	Senevirathna
Dilhani	
Dilanka	Ambepitiya
Binara	
Lasitha	Alponsu

Teacher

tchNo	tchName
1	Suranga
2	Nishan
3	Susith
4	Padma
5	Rukshan
6	Rathnasiri
7	Mohan

Department

depNo	depName	tchNo
1	Information Technology	1
2	Administration	3
3	Engineering	2
4	Languages	6

Project

proNo	proName	proLocation
1	Computer Lab	Galle
2	Wi-Fi	Gampaha
3	Computer Lab	Colombo

Supplier

supNo	supName
1	Cisco
2	HP
3	Intel
4	Dell
5	AMD
6	Sony

Supply

proNo	supNo	material	quantity
3	2	Desktop Computers	10
2	1	Routers	2
4	3	Network Servers	2
3	1	Proxy Server	1
1	3	Desktop Computers	8
3	6	DVD Writers	4

Order

odrNo	odrDate	odrPrice	customer
1	2008/11/12	1000	Sarath
2	2008/10/23	1600	Nimal
3	2008/09/02	700	Sarath
4	2008/09/02	300	Sarath
5	2008/08/30	2000	Kumara
6	2008/10/23	100	Nimal

Product

Product_ID	Unit_Price
1	10.34
2	12.78

Column Selecting (Projection)

Q1. All Columns

Query: `select * from Student;`

stuNo	stuName	gender	age	tpNo	city
1	Namal	male	22	0013456702	Gampaha
2	Kumara	male	27	0217896465	Boralla
3	Vishaka	female	25	0799956662	Boralla
4	Anuradha	male	23	0567856705	Nawala
5	Nirmala	female	18	0987034702	Gampaha
6	Amal	male	32	0348787890	Maharagama

Q2. Selected Columns -> Show stuNo & stuName

Query: `select stuNo, stuName from Student;`

stuNo	stuName
1	Namal
2	Kumara
3	Vishaka
4	Anuradha
5	Nirmala
6	Amal

Q3. Using Allies

Query: `Select stuNo as number, stuName as name from student;`

number	name
1	Namal
2	Kumara
3	Vishaka
4	Anuradha
5	Nirmala
6	Amal

Row Selecting (Selection)

The **where clause** is used for filtering out rows/records of a table that fulfil a specified **condition**.

E.g. Show **all details of female students** in student table

Query: `Select * from Student where gender = 'female';`

stuNo	stuName	gender	age	tpNo	city
3	Vishaka	female	25	0799956662	Boralla
5	Nirmala	female	18	0987034702	Gampaha

Operators are used in where clause. In above example equal operator (Comparison Operator) is used.

The value in the condition **requires single quotes or double around text values** whereas **numerical values should not be enclosed within quotes**.

Operators

1. Comparison Operators (= , != , <> , > ,>= ,>!, < ,<= ,<!)

Q5. Show **student number and name of female students** in student table

Query: Select **stuNo**, **stuName** from Student where **gender = 'female'**;

stuNo	stuName
3	Vishaka
5	Nirmala

Q6. Show **student number, name and telephone number of kumara** in student table.

Query: Select **stuNo** , **stuName** , **tpNo** from Student where **stuName = 'Kumara'**;

stuNo	stuName	tpNo
2	Kumara	0217896465

Q7. Show **student number, name and age** of students whose **age is 22** in student table

Query: Select **stuNo** , **stuName** , **age** from Student where **age = 22**;

stuNo	stuName	age
1	Namal	22

Q8. Show **student number, name and age** of students whose **age is greater than 22** in student table.

Query: Select **stuNo** , **stuName** , **age** from Student where **age >22**;

stuNo	stuName	age
2	Kumara	27
3	Vishaka	25
4	Anuradha	23
6	Amal	32

Q9. Show **student number, name and age** of students whose **age not 22** in student table.

Query 1: `select stuNo, stuName, age from student where age!=22;`

Query 2: `select stuNo, stuName, age from student where age<>22;`

stuNo	stuName	age
2	Kumara	27
3	Vishaka	25
4	Anuradha	23
5	Nirmala	18
6	Amal	32

2. Arithmetic Operators (+, -, *, /, %)

Consider employee table in page no.2.

Q10. Show first name, last name and salary in student table.

As you seen salary is not stored in the database. Calculate using following formulae.

$$\text{salary} = \frac{(\text{basicSalary} * \text{incentiveRate})}{100} + \text{vehicleAllownce} + \text{basicSalary}$$

Query: `Select firstname,`

`lastname,`

`((basicsalary/100)*incentiverate)+vehicleallownce+basicsalary) as salary`

`from employee;`

firstname	lastname	salary
Nimal	Perera	24500.00000000
Sunil	Silva	21350.00000000
Nimali	Sirisena	18375.00000000
Ruwan	Gamage	57000.00000000

3. SQL Logical Operators (NOT, AND, OR, IS NULL, LIKE, IN, EXISTS, BETWEEN, ANY,ALL)

Q11. Show **student number, name and age** of students whose **age not less than 22** in student table.

Query: `select stuNo, stuName, age from student where not age < 22;`

stuNo	stuName	age
1	Namal	22
2	Kumara	27
3	Vishaka	25
4	Anuradha	23
6	Amal	32

Q12. Show **student number, name and age** of **female** students whose **age is greater than 22** in student table.

Query: `Select stuName, age, gender from Student where age > 23 And gender='female';`

stuName	gender	age
Vishaka	female	25

Q13. Show **student name, gender and age** of **female** students **or age is greater than 22** in student table.

Query: `Select stuName, age, gender from Student where age > 23 Or gender = 'female';`

stuName	gender	age
Kumara	male	27
Vishaka	female	25
Nirmala	female	18
Amal	male	32

Q14 Show **student name, age and gender and city** of **male students** whose **age is greater than 25 or coming from Gampaha** city in student table.

Query: `Select stuName, age, gender, city
from student
where gender = 'male' And(age > 25 Or city = 'Gampaha');`

stuName	gender	age	city
Namal	male	22	Gampaha
Kumara	male	27	Boralla
Amal	male	32	Maharagama

Q15

Query: `Select stuNo, stuName, age from Student where age > All (22,25);`

stuNo	stuName	age
2	Kumara	27
6	Amal	32

Q16

Query: `Select stuNo, stuName, age from Student where age > Any (22,25);`

stuNo	stuName	age
2	Kumara	27
3	Vishaka	25
4	Anuradha	23
6	Amal	32

Q17

Query: `Select stuNo, stuName, tpNo
from Student
where stuName In ('Kumara' , 'Anuradha', 'Namal');`

stuNo	stuName	tpNo
1	Namal	0013456702
2	Kumara	0217896465
4	Anuradha	0567856705

Q18

Query: select stuNo, stuName, age from student where age between 22 and 25;

stuNo	stuName	age
1	Namal	22
3	Vishaka	25
4	Anuradha	23

Q19

Query: Select * from Visitors where lastName is Null ;

firstName	lastName
Dilhani	
Binara	

Nested Queries (Sub Queries)

E.g. Show the name of the teacher who works in the Information Technology department.

Query: `Select tchName
from Teacher
where tchNo = (Select tchNo
from Department
where depName = "Information Technology");`

tchName
Suranga

Outer Query

Should be evaluated finally. Because, it uses data/input values from the Inner Query.

Inner Query

Should be evaluated first. Outer Query need the data from the Inner Query

Important Inner Query could join in three ways depending on its output

If the output has

(i)	A single Value	=
(ii)	Multiple Values of the Same Column/Domain	In
(iii)	If it is unknown the Column Name, But know that there is a matching Column	Exists

Q20

Query: `Select tchName from Teacher where tchNo In (Select tchNo from Department);`

tchName
Suranga
Nishan
Susith
Rathnasiri

Q21

Query: `Select tchName
from Teacher
where Exists (
 Select * from Department where Department.tchNo = Teacher.tchNo
);`

tchName
Suranga
Nishan
Susith
Rathnasiri

Q22. Write output of the following SQL query.

Query: `Select supName
from Supplier
where supNo In (
 Select supNo from Supply where proNo = (
 Select proNo from Project where proLocation = 'Colombo'
)
);`

Q23. Write SQL query for below question.

Question : What are the Project Locations that the “Cisco” supply Equipment’s?

Nested Queries (Sub Queries)

2014-07 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	2014-28 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	2014-24 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	2014-25 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
2011-20 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	2011-21 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	2011-32 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	2017-31 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
2016-19 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	2015-08 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>		

DISTINCT Keyword

Q24

Query: Select distinct gender as genderNames from student;

genderNames

male

female



2011-30 ☐ ☐ ☐

Ordering (sorting)**Ascending**

Q25

Query: Select * from student order by age;

Select * from student order by age asc;

stuNo	stuName	gender	age	tpNo	city
5	Nirmala	female	18	0987034702	Gampaha
1	Namal	male	22	0013456702	Gampaha
4	Anuradha	male	23	0567856705	Nawala
3	Vishaka	female	25	0799956662	Boralla
2	Kumara	male	27	0217896465	Boralla
6	Amal	male	32	0348787890	Maharagama

Q26

Query: Select * from student order by gender, age;

stuNo	stuName	gender	age	tpNo	city
5	Nirmala	female	18	0987034702	Gampaha
3	Vishaka	female	25	0799956662	Boralla
1	Namal	male	22	0013456702	Gampaha
4	Anuradha	male	23	0567856705	Nawala
2	Kumara	male	27	0217896465	Boralla
6	Amal	male	32	0348787890	Maharagama

Descending

Q27

Query: Select * from student order by age desc;

stuNo	stuName	gender	age	tpNo	city
6	Amal	male	32	0348787890	Maharagama
2	Kumara	male	27	0217896465	Boralla
3	Vishaka	female	25	0799956662	Boralla
4	Anuradha	male	23	0567856705	Nawala
1	Namal	male	22	0013456702	Gampaha
5	Nirmala	female	18	0987034702	Gampaha

Q28

Query: Select * from student order by gender desc, age;

stuNo	stuName	gender	age	tpNo	city
1	Namal	male	22	0013456702	Gampaha
4	Anuradha	male	23	0567856705	Nawala
2	Kumara	male	27	0217896465	Boralla
6	Amal	male	32	0348787890	Maharagama
5	Nirmala	female	18	0987034702	Gampaha
3	Vishaka	female	25	0799956662	Boralla

pp Ordering (sorting)

2010-40 ☐ ☐ ☐2014-30 ☐ ☐ ☐2018-23 ☐ ☐ ☐2017-25 ☐ ☐ ☐

Database Functions

Aggregate Functions / Summarizing (Group Functions)

Q29

Query: Select Avg(odrPrice) as AveragePrice from Order ;

AveragePrice
950

Q30

Query: Select customer, odrPrice
 from Order
 where odrPrice > (Select Avg(odrPrice) from Order);

odrPrice	customer
1000	Sarath
1600	Nimal
2000	Kumara

Q31

Query: `Select Count(customer) As sarathAmount from Order where customer = 'Sarath';`

sarathAmount
3

Q32

Query: `Select Count(*) As orderAmount from Order ;`

orderAmount
6

Q33

Query: `Select Count(Distinct customer) As numberOfCustomers from Order;`

numberOfCustomers
3

Q34

Query: `Select First(odrPrice) As firstOrderPrice from Order;`

`Select odrPrice from Order Order By odrNo Limit 1; (MySQL Query)`

firstOrderPrice
1000

Q35

Query: `Select Last(odrPrice) As lastOrderPrice from Order;`
 `Select odrPrice from Order Order By odrNo Desc Limit 1;`

lastOrderPrice
100

Q36

Query: `Select Max(odrPrice) As largestOrderPrice from Order;`

largestOrderPrice
2000

Q37

Query: `Select Min(odrPrice) As smallestOrderPrice from Order;`

smallestOrderPrice
100

Q38

Query: `Select Sum(odrPrice) As totalOrderPrice from Order ;`

customer	Sum(odrPrice)
Sarath	5700
Nimal	5700
Sarath	5700
Sarath	5700
Kumara	5700
Nimal	5700

Q39

Query: `Select customer, Sum(odrPrice) from Order Group By customer ;`

customer	Sum(odrPrice)
Sarath	2000
Nimal	1700
Kumara	2000

Q40

Query: `Select customer, Sum(odrPrice)`
 `from Order`
 `Where customer = 'Sarath' Or customer ='Kumara'`
 `Group By customer`
 `having Sum(odrPrice) > 1500 ;`

customer	Sum(odrPrice)
Sarath	2000
Kumara	2000

Q41

Query: `Select customer ,odrDate ,Sum(odrPrice)`
 `from Order`
 `Group By customer, odrDate ;`

customer	odrDate	Sum(odrPrice)
Sarath	2008/11/12	1000
Sarath	2008/09/02	1000
Nimal	2008/10/23	1700
Kumara	2008/08/30	2000

Scalar Functions

Q42

Query: `Select Ucase(customer) As name from Order ;`

name
SARATH
NIMAL
SARATH
SARATH
KUMARA
NIMAL

Q43

Query: `Select odrNo, Lcase(customer) As name from Order ;`

odrNo	name
1	sarath
2	nimal
3	sarath
4	sarath
5	kumara
6	nimal

Q44

Query: `Select odrNo, Mid(customer,2,3) As name from Order;`

odrNo	name
1	ara
2	ima
3	ara
4	ara
5	uma
6	ima

Q45

Query: `Select odrNo , odrDate , Now() As today from Order ;`

odrNo	odrDate	today
1	2008/11/12	12/02/2009 11.25.34 AM
2	2008/10/23	12/02/2009 11.25.34 AM
3	2008/09/02	12/02/2009 11.25.34 AM
4	2008/09/02	12/02/2009 11.25.34 AM
5	2008/08/30	12/02/2009 11.25.34 AM
6	2008/10/23	12/02/2009 11.25.34 AM

Q46

Query: Select odrNo , odrDate , Format(Now() , 'YYYY-MM-DD') As today
 from Order ;

odrNo	odrDate	today
1	2008/11/12	12-02-2009
2	2008/10/23	12-02-2009
3	2008/09/02	12-02-2009
4	2008/09/02	12-02-2009
5	2008/08/30	12-02-2009
6	2008/10/23	12-02-2009

Q47

Query: Select Product_ID, Round(Unit_Price, 1) As Price from Product ;

Product_ID	Price
1	10.3
2	12.8



Database Functions

2013-18 ☐ ☐ ☐2018-17 ☐ ☐ ☐2017-33 ☐ ☐ ☐2016-20 ☐ ☐ ☐2017-14 ☐ ☐ ☐

Grouping

Q48

Query: `Select city, count(city) from Student Group By city ;`

city	count(city)
Gampaha	2
Boralla	2
Nawala	1
Maharagama	1

Q49

Query: `Select city As Area, count(city) As Amount from Student Group By city;`

Area	Amount
Gampaha	2
Boralla	2
Nawala	1
Maharagama	1

Having Keyword

Q50

Query: `Select city As Area, count(city) As Amount`
 `from Student`
 `Group By city`
 `Having count(city) > 1`

Area	Amount
Gampaha	2
Boralla	2

Q51

Query: Select city As Area, count(city) As Amount
 from Student
 Group By city
 Having count(city) > 1
 Order By city;

Area	Amount
Boralla	2
Gampaha	2

Priority Order



2014 – 23 ☐ ☐ ☐



Grouping

2015-09 ☐ ☐ ☐
 2016-16 ☐ ☐ ☐
 2015-16 ☐ ☐ ☐

2015-10 ☐ ☐ ☐
 2016-18 ☐ ☐ ☐

2012-32 ☐ ☐ ☐
 2016-21 ☐ ☐ ☐

2011-29 ☐ ☐ ☐
 2016-22 ☐ ☐ ☐

Wild Card Characters

%

Zero or More Characters

–

Exact Number of Characters

[ChaList]

Single Character from a given List

[!ChaList], [^ChaList]

Except any Character from a given List

Q52

Query: Select * from Visitors where lastName like 'Wi%' ;

firstName	lastName
Nimal	Wimalachandra
Suren	Wikramarachchi

Q53

Query: Select * from Visitors where firstName like 'Su _ _ _' ;

firstName	lastName
Sunil	Basnayake
Suren	Wikramarachchi

Q54

Query: Select * from Visitors where lastName like '%pitiya' ;

firstName	lastName
Namal	Manampitiya
Dilanka	Ambepitiya

Q55

Query: Select * from Visitors where firstName like '[ND]%' ;

firstName	lastName
Nimal	Wimalachandra
Namal	Manampitiya
Dulanga	Senevirathna
Dilhani	
Dilanka	Ambepitiya

Q56

Query: Select * from Visitors where lastName like '%[!vp]%' ;

firstName	lastName
Nimal	Wimalachandra
Sunil	Basnayake
Suren	Wikramarachchi
Dilhani	
Binara	

LIMIT and TOP Keyword

Q57

Query: `Select * from Visitors Limit 5 ;`

firstName	lastName
Nimal	Wimalachandra
Sunil	Basnayake
Suren	Wikramarachchi
Namal	Manampitiya
Dulanga	Senevirathna

Q58

Query: `Select Top 3 from Visitors ;`

firstName	lastName
Nimal	Wimalachandra
Sunil	Basnayake
Suren	Wikramarachchi

Q59

Query: `Select Top 50 Percent from Visitors ;`

firstName	lastName
Nimal	Wimalachandra
Sunil	Basnayake
Suren	Wikramarachchi
Namal	Manampitiya
Dulanga	Senevirathna

Joins

Consider following tables

Employee

employeeNo	employeeName	departmentId
1901	Kamal	01
1902	Sunil	02
1903	Nimal	01
1904	Bimal	01
1905	Nayana	Null

Department

departmentId	departmentName
01	Information Technology
02	Human Resources
03	Administration

Cross Join (Like Cartesian Product)

This Query will combine each row of the first table with each row from second table.

Query: Select * from employee CROSS JOIN department; (explicit query)
 Select * from employee, department; (implicit query)

Output: Rows rows of employee x department

5 x 3

15

Columns columns of employee + columns of department

3 + 2

5

employeeNo	employeeName	departmentId	departmentId	departmentName
1901	Kamal	01	01	Information Technology
1901	Kamal	01	02	Human Resources
1901	Kamal	01	03	Administration
1902	Sunil	02	01	Information Technology
1902	Sunil	02	02	Human Resources
1902	Sunil	02	03	Administration
1903	Nimal	01	01	Information Technology
1903	Nimal	01	02	Human Resources
1903	Nimal	01	03	Administration
1904	Bimal	01	01	Information Technology
1904	Bimal	01	02	Human Resources
1904	Bimal	01	03	Administration
1905	Nayana	Null	01	Information Technology
1905	Nayana	Null	02	Human Resources
1905	Nayana	Null	03	Administration



Cross join

2015-18 ☐ ☐ ☐

2014-19 ☐ ☐ ☐

2015-09 ☐ ☐ ☐ 2015-16 ☐ ☐ ☐

This operation requires two matching columns in a joined table.

```
Query:      Select *                                (explicit query)
            from employee
              inner join department
                on employee.departmentId = department.departmentId;
```

```
Select *                                     (implicit query)
  from employee, department
  where employee.departmentId = department.departmentId
```

departmentId	departmentName
01	Information Technology
02	Human Resources
03	Administration

employeeNo	employeeName	departmentId	departmentId	departmentName
1901	Kamal	01	01	Information Technology
1901	Kamal	01	02	Human Resources
1901	Kamal	01	03	Administration
1902	Sunil	02	01	Information Technology
1902	Sunil	02	02	Human Resources
1902	Sunil	02	03	Administration
1903	Nimal	01	01	Information Technology
1903	Nimal	01	02	Human Resources
1903	Nimal	01	03	Administration
1904	Bimal	01	01	Information Technology
1904	Bimal	01	02	Human Resources
1904	Bimal	01	03	Administration
1905	Nayana	Null	01	Information Technology
1905	Nayana	Null	02	Human Resources
1905	Nayana	Null	03	Administration

employeeNo	employeeName	departmentId	departmentId	departmentName
1901	Kamal	01	01	Information Technology
1902	Sunil	02	02	Human Resources
1903	Nimal	01	01	Information Technology
1904	Bimal	01	01	Information Technology

Hence the join predicate is equality these joins are also called **equi-join**.



Inner join

2010-28 ☐ ☐ ☐ 2014-20 ☐ ☐ ☐ 2017-31 ☐ ☐ ☐ 2013-23 ☐ ☐ ☐

2011-26 ☐ ☐ ☐ 2015-14 ☐ ☐ ☐

2016-16 ☐ ☐ ☐ 2016-18 ☐ ☐ ☐ 2010-30 ☐ ☐ ☐ 2011-20 ☐ ☐ ☐

2011-21 ☐ ☐ ☐ 2013-24 ☐ ☐ ☐ 2012-24 ☐ ☐ ☐ 2012-25 ☐ ☐ ☐

2013-21 ☐ ☐ ☐ 2014-29 ☐ ☐ ☐

Natural Join

Natural join is a special case of equi-join. Natural join will show one of the common columns in the result table. Also, the matching columns must be in same name.

Query: `select * from employee natural join department;`

employeeNo	employeeName	departmentId
1901	Kamal	01
1902	Sunil	02
1903	Nimal	01
1904	Bimal	01
1905	Nayana	Null

departmentId	departmentName
01	Information Technology
02	Human Resources
03	Administration

* Only one of the common columns in the result.

departmentId	employeeNo	employeeName	departmentName
01	1901	Kamal	Information Technology
02	1902	Sunil	Human Resources
01	1903	Nimal	Information Technology
01	1904	Bimal	Information Technology



Natural Join

2018-28 ☐ ☐ ☐ 2017-32 ☐ ☐ ☐

Outer Join

Left outer join

Query: Select *
 from employee **left outer join** department
 ON employee.departmentId = department.departmentId;

employeeNo	employeeName	departmentId
1901	Kamal	01
1902	Sunil	02
1903	Nimal	01
1904	Bimal	01
1905	Nayana	Null

departmentId	departmentName
01	Information Technology
02	Human Resources
03	Administration

*every record in employee is in the result

employeeNo	employeeName	departmentId	departmentId	departmentName
1901	Kamal	01	01	Information Technology
1902	Sunil	02	02	Human Resources
1903	Nimal	01	01	Information Technology
1904	Bimal	01	01	Information Technology
1905	Nayana	Null	Null	Null



left outer join

2018-39 ☐ ☐ ☐

2015-08 ☐ ☐ ☐ 2013-20 ☐ ☐ ☐

Right outer join

Query: Select *
 from employee **right outer join** department
 ON employee.departmentId = department.departmentId;

employeeNo	employeeName	departmentId
1901	Kamal	01
1902	Sunil	02
1903	Nimal	01
1904	Bimal	01
1905	Nayana	Null

departmentId	departmentName
01	Information Technology
02	Human Resources
03	Administration

*every record in department is in the result

employeeNo	employeeName	departmentId	departmentId	departmentName
1901	Kamal	01	01	Information Technology
1902	Sunil	02	02	Human Resources
1903	Nimal	01	01	Information Technology
1904	Bimal	01	01	Information Technology
Null	Null	Null	03	Administration

Right Outer join

2017-44 ☐ ☐ ☐

2010-23 ☐ ☐ ☐ 2014-22 ☐ ☐ ☐

2016-16 ☐ ☐ ☐ 2013-19 ☐ ☐ ☐

Full outer join (Not Support in MySQL)

Query: Select *
 from employee **full outer join** department
 ON employee.departmentId = department.departmentId;

employeeNo	employeeName	departmentId
1901	Kamal	01
1902	Sunil	02
1903	Nimal	01
1904	Bimal	01
1905	Nayana	Null

departmentId	departmentName
01	Information Technology
02	Human Resources
03	Administration

*every record in employee is in the result

*every record in department is in the result

employeeNo	employeeName	departmentId	departmentId	departmentName
1901	Kamal	01	01	Information Technology
1902	Sunil	02	02	Human Resources
1903	Nimal	01	01	Information Technology
1904	Bimal	01	01	Information Technology
1905	Nayana	Null	Null	Null
Null	Null	Null	03	Administration

Insert

Consider following meta table

student

Field Name	Data Type	Length	Constraints
id	int	5	Primary Key
name	varchar	200	
age	int	2	

Single Insert

Query 1: Insert into student values(1,'Niroshan',22);

Query 2: Insert into student(id, name, age) values(2,'Kamal',23);

Query 3: Insert into student(id, name) values(3,'Sandun');

Multiple Insert

Query 1: Insert into student values(4,'Sahan',23),(5,'Prabath',25);

Query 2: Insert into student(id, name, age) values (6,'Nimal',null),
(7,'Sunil',25);

Query 3: Insert into student(id, name) values(8,'Ruwan'), (9,'Sunimal');



Insert Query

2015-07 ☐ ☐ ☐ 2014-21 ☐ ☐ ☐ 2011-27 ☐ ☐ ☐ 2011-31 ☐ ☐ ☐

2011-40 ☐ ☐ ☐ 2010-19 ☐ ☐ ☐ 2010-20 ☐ ☐ ☐ 2009-25 ☐ ☐ ☐

2007-24 ☐ ☐ ☐ 2006-23 ☐ ☐ ☐

Update

All Rows Update

E.g. Updates age as 22 on every record.

Query: Update student set age = 22;

Selecting Row Update (Where clause)

E.g. Updates age as 22 where value is null.

Query: Update student set age = 22 where age is null;



Update Query

2017-27	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	2013-39	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	2012-26	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	2012-28	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2009-27	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	2007-21	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	2006-31	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	2005-31	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Delete

All Rows Delete

Using Delete Keyword

Delete from student;

Using Truncate Keyword

Truncate table student;

Selecting Row Delete

E.g. Deletes students whose age is greater than 23.

Query: Delete from student where age>23;



Delete Query

2016-25 ☐ ☐ ☐

2013 – 25 ☐ ☐ ☐

2011 – 22 ☐ ☐ ☐

2010-24 ☐ ☐ ☐

2009-29 ☐ ☐ ☐

2008 – 10 ☐ ☐ ☐
