SQL Course

Date / /
Page No.

Structure Query Language.

☆ Table of Contents.

- Introduction to DBMS & SQL
- Normalization
- Subsets of SQL
  - DDL
  - DML
  - DCL
  - TCL
- SQL Operators
- SQL Functions
- SQL Joins
- Lab Session (Projects)

My SQL Notes

Scanned with CamScanner

Date / /
Page No.

☆ What is Database?

Database is a Collection of information organized for easy access, management and maintenance.

- Example:
  - Telephone directory
  - Customer data
  - Product inventory
  - Visitors' register
  - Weather records

☆ Types of Data Models

- Record based logical model
  - Hierarchical data model
  - Network data model
  - Relational data model
- Object based logical model
  - Entity relationship model

Scanned with CamScanner

Date / /
Page No.

DBMS operations

Adding new files (Tables)
Inserting data
Retrieving data
Modifying data
Removing data
Removing files

Web application बनाओ Jango.

Advantages of DBMS

Sharing of data across applications
Enhanced security mechanism
Enforce integrity Constraints
Better transaction support
Backup and recovery features.

Scanned with CamScanner

Introduction to RDBMS.

- A relational database refers to a database that stores data in a structured format, using rows and columns.
- This makes it easier to locate and access specific values within the database.
- It is "relational" because the values within each table are related to each other. Tables may also be related to other tables.
- The relational structure makes it possible to run queries across multiple tables at once.

Features of RDBMS

- Every piece of information is stored in form of tables
- Has primary keys for unique identification of rows
- Has foreign keys to ensure data integrity.
- Provides SQL for data access.
- Uses indexes for faster data retrieval
- Gives access privileges to ensure data security.

Scanned with CamScanner

Date / /
Page No.

Normalization

- Decomposes large, complex table into simpler and smaller ones.
- Moves from lower normal forms to higher normal forms.

Normal forms

First Normal form (1NF) | 2NF | 3NF | BCNF, 4NF, 5NF

Need for normalization

- In order to produce good database design.
- To ensure all database operations to be efficiently performed
- Avoid any expensive DBMS operations.
- Avoid unnecessary replication of information

Student_Details			Course_details				Pre-requisite	Result_details		
101	Jack	11/4/1975	M1	Advance Math	17		Basic Math	03/11/2015	82	
102	Rock	10/4/1976	P4	" Physics	18		" physics	22/11/2015	83	
103	Mary	11/7/1975	B3	" Biology	10		" Biology	14/11/2015	58	
04	Roby	10/4/1976	H6	" History	19		" History	22/11/2015	83	
05	Jim	3/8/1978	C3	" Chemistry	12		" Biology	15/11/2015	50	C

Raw database

Scanned with CamScanner

Functional Dependency.

- Consider the relation
  - Result (student#, Course#, CourseName#, marks#, Grade#)
    - Student# and Course# together define exactly one value of marks. Student#, Course#, marks
    - Student# and Course# determines Marks or Marks is functionally dependent on Student# and Course#.

- other functional dependencies in the relation:
  - Course# – CourseName
  - Marks# – Grade

- In a given relation R, P and Q are attributes. Attribute Q is functionally dependent on attribute P if each value of P determines exactly one value of Q.

$$\boxed{P} \longrightarrow \boxed{Q}$$

Scanned with CamScanner

Date / /
Page No.

Functional dependency Types

- Partial functional dependency.
- Transitive dependency.

Partial Functional Dependency.

- Attribute Q is partially dependent on attribute P, if and only if it is dependent on the subset of attribute P.
- REPORT (Student#, Course#, StudentName, CourseName, Marks, Grade)

$$\boxed{\text{Student\#, Course\#}} \longrightarrow \boxed{\text{Marks}}$$

$$\boxed{\text{Course\#}} \longrightarrow \boxed{\text{CourseName}}$$

Transitive Dependency.

X, Y, Z are three attributes

$$\boxed{X} \rightarrow \boxed{Y} \qquad \boxed{Y} \rightarrow \boxed{Z} \qquad \boxed{X} \rightarrow \boxed{Z}$$

$$\boxed{\text{Student\#}} \rightarrow \boxed{\text{Course\#}} \rightarrow \boxed{\text{Marks\#}} \rightarrow \boxed{\text{Grade\#}}$$

Scanned with CamScanner

Date / /
Page No.

☆ 1NF

A relation Schema is in 1NF, if and only if:

→ All Attributes in the relation are atomic (indivisible Value)

→ And there are no repeating elements or group of elements.

Student #	Student name	Date of Birth	Course#	Course name	Pre-Requisite	Duration in days	Date of exam	Marks	Grade
101	Jack	11/4/1975	M1	Advance math	Basic math	17	02/11/2015	82	A
102	Roby	10/4/1976	P4	Advance physics	Basic physics	18	21/11/2015	83	A
103	Mary	11/7/1975	B3	Advance Biology	Basic Biology	10	2012/11/2015	68	B
...	...	...	...	...	...	...	...	...	...

1NF Student Table

☆ 2NF

→ It is in 1st Normal form.

→ No partial dependency exists between non-key attributes and key attributes.

- Student#, Course# → Marks
- Student#, Course# → Grade
- Marks → Grade

- Student# → Student Name, DOB
- Course# → Course Name, Pre-Requisite, Duration Days, Date of Exam

→

Scanned with CamScanner

Date / /
Page No.

Partial Dependency with the key attribute → Split/ Decompose the tables to remove partial dependencies.

2NF

Student table.

Student#	Student_Name	Date of Birth
101	Jack	11/4/1975
102	Roby	10/4/1976
103	Mary	11/7/1978

Result table

Student#	Course#	Marks	Grade
101	M1	82	A
102	P4	83	A
103	B3	68	B

Course Table

Course#	Course Name	Prerequisite	Duration in days	date of Exam.
M1	Advance Maths	Basic Maths	17	02/11/2015
P4	" Physics	" physics	18	21/11/2015
B3	" Biology	" Biology	10	12/4/2015

Scanned with CamScanner

3NF

- It is in 2NF
- No transitive dependency exists between non-key attributes and key attributes through another non-key attribute.

Result_table

Student#	Course#	Marks	Grade
101	M1	82	A
102	P4	83	A
103	B3	68	B

- Student#, Course# → Marks
- Student#, Course# → Grade
  Marks → Grade

$\boxed{\text{Student\#, Course\#} \rightarrow \text{Marks} \rightarrow \text{Grade : TD}}$

→ remove

Result table

Student#	Course#	Marks
101	M1	82
102	P4	83
103	B3	68

Marks Grade Table

Marks	Grade
82	A
83	A
68	B

Scanned with CamScanner

SQL

Date / /
Page No.

- Programming language specially designed for working with database to...

- CREATE
- MANIPULATE
- SHARE/ACCESS

Why?

- Allows user to Communicate, i.e, access and manipulate the database.
- Allows user to retrieve data from a database.
- Allows users to create, update, modify and delete the database.

SQL is a language for defining the structure of a database.

SQL Data Types

1) Numeric - bit, tinyint, smallint, int, bigint, decimal, numeric, float, real.
2) Character/String :- char, Varchar, Text
3) Date/Time :- Date, Time, Datetime, Timestamp, Year
4) Miscellaneous :- Json, XML

Scanned with CamScanner

SQL Constraints.

NOT Null	Ensures that a Column does not have a Null Value.
Default	Provides a default value for a Colum when none is Specified.
Unique	Ensures that all the Values in Colu are different.
Primary	Identifies each row/record in a datab -ase table Uniquely.
Check	Ensures that all Values in a Colum satisfy Certain Conditions.
Index	Creates and retrieves data from the database very quickly.

SQL Subsets.

* **DDL**

Create	Creates objects in the database objects.
Alter	Alters the Structure of the database/dB object.
Drop	Deletes objects from the db
Truncate	Removes all records from table permanently
Rename	Renames an object.

Scanned with CamScanner

Date / /
Page No.

- Create Command

```
CREATE TABLE employees (
    emp_id INT(10) NOT NULL,
    first_name VARCHAR(20),
    last_name VARCHAR(20) NOT NULL,
    Salary int(10) NOT NULL,
    PRIMARY KEY (emp_id));
```

emp_id	First_name	last_name	Salary

~~Sho~~ To Show Structure of table
Describe employees;

- Alter Command (Modify)
(To add Column)

Alter Table employees ADD Column Contact INT(10);

- Rename Command

Alter table employees RENAME COLUMN contact TO job_code;

- Truncate Command
(To remove Values)

Truncate table employees;

Scanned with CamScanner

Date / /
Page No.

- Drop Command:-
  (Delete ~~Column or~~ full Table)

- Drop Table table_name;
- Drop Table employees;

☆ DML

Insert	Insert data into a Table
Update	Updates existing data within a Table
Delete	Deletes specified/all records from a table

- Insert Command.

Insert into employees
(emp-id, first_name, last_name, salary) Values
(101, 'Steven', 'King', 10000);

Insert into employees
(emp-id, first_name, last_name, salary) Values
(102, 'Edwin', 'Thomas', 15000);

Insert into employees
(emp-id, first-name, last-name, salary) Values
(103, 'Harry', 'Potter', 20000);

Scanned with CamScanner

Date / /
Page No.

- Update Command
  (To update any Value)

Update employees
set last_name = 'cohen'
where emp_id = 101;

- Delete Command
  (To Delete a record)

Delete from employees where emp_id IN(101, 103);

DCL

Grant — Gives access privelege to database
Revoke — withdraws access privileges given with the grant Command

Grant <Privilege list> on <relation name> to <user>

Revoke <Privilege list> on <relation name> to <user>

TCL
(Transaction Control)

Commit — Saves the work done
Rollback — Restores database to origin state since the last Commit.
Savepoint — Identify a point in a transaction to which you can roll back on.

Scanned with CamScanner

SQL Operators :-

Filter

WHERE Clause :-

- Used to Specify a Condition while fetching the data from a Single table or by joining with multiple tables.
- Not only used in the Select Statement, but it is also used in UPDATE, DELETE Statement etc.

e.g.:

```
Select * from employees where emp_id=101;
```

It will extracts all the Columns from the table 'employees' whose emp_id = 101

Logical :-

Operator	illustrative Example	Result
AND	(S<2) AND (S>3)	FALSE
OR	(S<2) OR (S>3)	TRUE
NOT	NOT (S<2)	TRUE

Sample Queries :

```
Select * from employees where first_name = 'steven' and
salary =15000;
" " " " first_name = 'steven' OR salary= 15000;
" " " " first_name = 'steven' and salary != 10000;
```

Scanned with CamScanner

~~Not~~

Date / /
Page No.

Comparison.

=	equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<> or !=	Not equal to

Sample Query:

- Select * from employees where first_name = 'Steven' and Salary <= 1000;

- Select * from employees where first_name = 'Steven' or Salary >= 10000;

- Select * from employees where first_name != 'Charlie';

Scanned with CamScanner

Special :-

Between	Checks an attribute Value within range
LIKE	checks an attribute Value matches a given string pattern
IS NULL	Checks an attribute Value is NULL
IN	checks an attribute Value matches any Value within a Value list
DISTINCT	Limits Values to Unique Values

Sample Queries:-

- Select * from employees where Salary between 10000 and 20000;
- Select * from employees where first_name like 'Steven';
- Select * from employees where Salary is null;
- Select * from employees where Salary in (10000, 12000, 20000);
- Select distinct(first_name) from employees;

like 'l%)
%l

agal 2 words ko Band kar
2%p

Aggregations.

Date / /
Page No.

Aggregation Functions.

Avg() Returns the average value from Specified Columns.
Count() Returns number of table rows
Max() Returns largest Value among the records
Min() Returns Smallest Value among the records
Sum() Returns the Sum of Specified Column Values.

Sample Queries.

- Select avg(Salary) from employees;
  Select Count(*) from employees;
  Select min(Salary) from employees;
  Select max(Salary) from employees;
  Select Sum(Salary) from employees;

We Can't use aggregate Function with 'WHERE'

Scanned with CamScanner

Date / /
Page No.

SQL Group By Clause.

- Arrange identical data into groups.

e.g.

```
select max(salary), dept_id from employees
Group by dept_id
```

SQL Having Clause

- Used with aggregate functions due to its non-performance in the WHERE clause.
- Must follow the GROUP BY clause in a query and must also precede the ORDER BY clause if used.

e.g;

```
Select avg(Salary), dept_id
from employees
Group by dept_id
Having Count(dept-id) >=2
```

Scanned with CamScanner

Date / /
Page No.

SQL ORDER BY Clause

- Used to sort output of SELECT statement
- default is to sort in ASC (Ascending)
- Can sort in Reverse (Descending) order with "DESC" after the Column name

e.g Select * from employees Order by Salary Desc;

SQL UNION

- Used to Combine the result-set of two or more SELECT statements removing duplicates
- Each SELECT statement within the UNION must have the same number of Columns.
- The Selected Columns must be of similar data types and must be in the same order in each SELECT statement.
- More than two queries Can be clubbed using more than one UNION statement.

Bimit

Scanned with CamScanner

SQL UNION

Product 1

Category-ID	Product Name
1	Nokia
2	Samsung
3	HP
6	Nikon

Product 2

CategoryID	Product_name
1	Samsung
2	LG
3	HP
5	Dell
6	Apple
10	Playstation

e.g

```
Select product_name from product1
UNION
select product_name from product2
```

Output :- It will not repeat Doblicate names

SQL Union ALL

- Used to Combine the results of two SELECT statements including duplicate rows.
- The same rules that apply to the UNION clause will apply to the UNION ALL operator.

Syntax :-

```
Select Col1, Col2 .. From table1
UNION ALL
Select Col1, Col2 .. From table2.
```

Scanned with CamScanner

Output :- It will write all names including Duplicates.

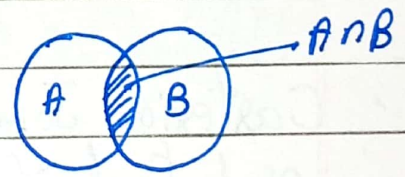
Intersection and others from PDF :-

* Sem 4 >> rdbms >> relational algebra basics

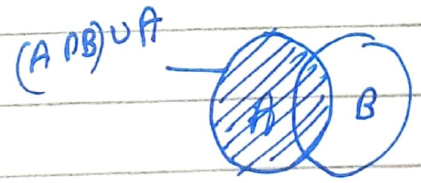
SQL Joins

Combine rows/columns from two or more tables, based on a related column b/w them in a database.

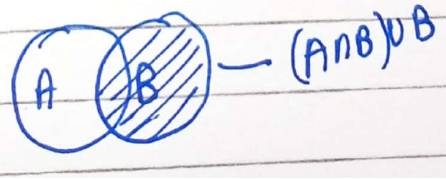
- INNER JOIN :- Returns rows when there is a match in both tables.



- LEFT JOIN :- Returns all rows from left table, even if there are no matches in the right table.



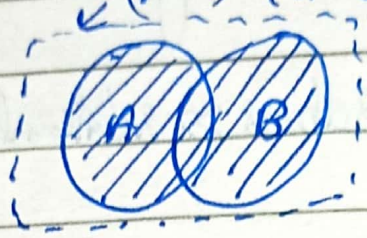
- Right Join :- Return all rows from the right table, even if there are no matches in the left table.



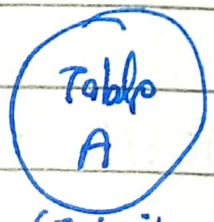
Scanned with CamScanner

- Full Outer Join :- Returns rows when there is a match in one of the tables.

$(A \cap B) \cup (A - B) \cup (B - A)$

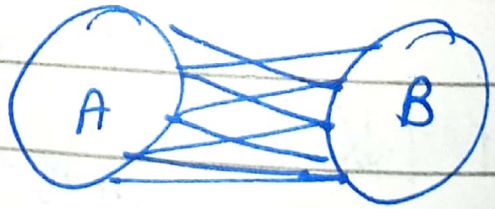


- Self Join :- Used to join a table to itself as if the table were two tables, temporarily renaming at least one table in the SQL Statement.



(Join itself)

- Cartesian Join (Cross Join) :- Return the Cartesian product of the sets of records from the two or more joined tables.



Scanned with CamScanner

SQL Inner Join

The Inner Join Creates a new result table by Combining Column Values of two tables (table1 and table2) based upon the join-predicate. The query Compares each row of table1 and each row of table2 to find all pairs of rows which satisfy the join-predicate.

Syntax:

```
Select table1.*Col1, table2.col2,..., table1.coln
from table1
Inner Join table2
ON table1. Commonfield = table2. Commonfield;
```

Employees

emp_id	first_name	last_name	Salary	dept_id
103	Harry	Potter	20000	12
102	Edwin	Thomas	15000	11
101	Steven	Cohen	10000	10
100	Erik	John	10000	12

Scanned with CamScanner

departments

dept_id	dept_name	manager_id	location_id
10	IT	200	1700
11	Marketing	201	1800
13	Resources	203	2400
14	Shipping	121	1500.

eg. Select employees.emp_id, employees.first_name, employees.last_name, departments.dept_id, departments.dept_name.

```
From employees
INNER JOIN departments ON
employees.dept_id = departments.dept_id;
```

emp_id	first_name	last_name	dept_id	dept_name
101	Steven	Cohen	10	IT
102	Edwin	Thomas	11	marketing

Scanned with CamScanner

Date / /
Page No.

SQL Left Join :-

The Left Join returns all the values from the left table, plus matched values from the right table or Null in case of no matching join predicate.

Syntax :-

```
Select Table1.Col1, Table2.Col2, ..., Table1.Coln
from Table1
Left Join Table2
On Table1.Commonfield = Table2.Commonfield;
```

SQL Right Join :-

```
Select e.emp_id, e.first_name, e.last_name, d.dept_
id, d.dept_name
from employees e
Right Join departments d
on e.dept_id = d.dept_id;
```

Right table departments

Output

emp_id	first_name	last_name	dept_id	dept_name
101	Steven	Cohen	10	IT
102	Edwin	Thomas	11	Marketing
Null	Null	Null	13	Resources
Null	Null	Null	14	Shipping

Scanned with CamScanner

Full Outer Join

The Full Outer Join Combines the results of both left and right outer joins. The joined table will contain all records from both the tables and fill in NULLs for missing matches on either side.

Syntax:

Left Join

```
Select table1.col1, table2.col2, ..., table1.coln
from table1
Left Join table2
on table1.commonfield = table2.commonfield;
```

Union

Right Join

```
Select table1.col1, table2.col2, ..., table1.coln
from table1
Right Join table2
On table1.commonfield = table2.commonfield;
```

Scanned with CamScanner

DATE / /
PAGE No.

Cross Join :-

The Cross Join produces a result set with the number of rows in the first table multiplied by the number of rows in the second.

Syntax:

```
Select table1.Col1, table2.Col2,..., Table1.coln
from table1
Cross Join table2;
```

- Assume there are 4 records in table1 and 3 records in table 2

```
Select * from table1
Cross JOIN table2;
```

table-1

alpha
A
B
C
D

table-2

Num
1
2
3

A	1
A	2
A	3
B	1
B	2
B	3
C	1
C	2
C	3
D	1
D	2
D	3

Scanned with CamScanner

Lab

DATE / /
PAGE No.

Datasets Used: Cricket_1.csv, Cricket_2.csv
Cricket_1 is the table for cricket test match 1.
Cricket 2 is the table for cricket test match 2

Q1) Find all the players who were present in the test match 1 or test match 2;

Q2) Write a MySQL query to find the players from the test match 1 having popularity higher than the average popularity.

Q3) Find player_id and player name that are common in the test match 1 and test match 2.

Q4) Retrieve player_id, runs and player_name from Cricket_1 table and display list of the players where the runs are more than the average runs.

Q5) Write a query to extract the player_id, runs and player_name from the Table "Cricket_1" where the runs are greater than 50.

Q6) Write a query to extract all the columns from Cricket_1 where player_name starts with 'S' and ends with 'v'.

Scanned with CamScanner

Q7) Write a query to extract all the Colomns from Cricket_1 where player_name does not end with 't'.

In Excel.

Cricket ~~Student~~_1.csv

Player_id	Player_Name	Runs	Popularity
PL1	Virat	50	10
PL2	Rohit	41	7
PL3	Jadega	33	6
PL4	Dhoni	35	15
PL5	Dhawan	45	12
PL6	Yadav	66	10
PL7	Raina	32	9
PL8	Binny	44	11
PL9	Rayudu	63	12
PL10	Rahane	21	10
PL11	A.patel	12	9
PL12	B.Kumar	30	7

Scanned with CamScanner

Cricket -

~~Student2.csv~~

Player_id	Player_Name	Runs	Charisma
PL1	Virat	50	55
PL2	Rohit	41	30
PL3	Jadeja	33	24
PL4	Dhoni	35	59
PL13	Yuvraj	33	50
PL6	Yadav	66	35
PL14	Tendulkar	74	80
PL8	Binny	44	32
PL9	Rayudu	63	39
PL15	Pravid	65	55
PL16	Yusuf	40	40

Scanned with CamScanner

Date / /
Page No.

Ans.

Q1) Select * from Cricket_1
union
Select * from Cricket_2;

OR Select Player_name from Cricket_1
Union
Select Player_name from Cricket_2;

Ans 2 Select player_name, Popularity from cricket_1 where Popularity > (Select AVG(popularity) from cricket_1);

Ans 3 Select player_id, player_name from cricket_1 where cricket_1.player_id IN (Select player_id from cricket_2);

Ans 4 Select player_id, runs, player_name from cricket_1 where runs > (Select AVG(runs) from cricket_1);

Ans 5 Select player_id, runs, player_name from cricket_1 where runs > 50;

Ans 6 Select * from Cricket_1 where player_name Like 'y%v';

Ans 7 Select * from Cricket_1 where player_name NOT Like '%t';

Scanned with CamScanner

Date / /
Page No.

Dataset new.

new_cricket.csv
(Combination of Both)

Q1) Extract the Player_id and Player_name of the players where the charisma Value is null.

Sol Select Player_id, Player_name from new_cricket where charisma is null;

Q2) Separate all player_id into single numeric id (example PL1 = 1).

Sol Select Player_id, SUBSTR(Player_Id, 3) from new_cricket;

Substr (Colomn name, Index)
↓
to extract

Q3) Write a MySQL query to extract Player_id, Player_name and Charisma where the charisma is greater than 25.

Sol Select player_id, Player_name, charisma from new_cricket where charisma > 25;

Scanned with CamScanner

Part 1

Date / /
Page No.

Question 1:

1) Create a Database bank

/* Solution */

```
create Database bank;      (Create database)
use bank;                  (Use it to create tables)
```

Question 2:-

2) Create a table with the name "bank details" with the following Columns

- Product with String data type
- Quantity with numerical data type
- Price with real number data type
- purchase cost with decimal data type
- estimated sale price with data type float.

/* Solution */

```
Create table bank details(
Product Char(10),
quantity Int,
price Real,
purchase_cost decimal(6,2),
estimated_sale_price Float);
```

maximum length = 6
After decimal can store two values

Scanned with CamScanner

Question 3)
#3) Display all Columns and their datatype and size in Bank details.

/* Solution */
Describe bank details;

Question 4)
#4) Insert two records in Bank details
--1st record will Values --
- product : PayCard
  Quantity : 3
  price : 330
  purchase cost : 8008
  estimated Sal_price : 9009
- Product : PayPoints
  Quantity : 4
  price : 200
  purchase cost : 8000
  estimated Sale price : 6800

/* Solution */
Insert into Bank details Values ('paycard', 3, 330, 8008, 9009);
Insert into Bank_details Values ('paypoints', 4, 200, 8000, 6800);

Scanned with CamScanner

Date / /
Page No.

Ques 5) Add a Column: Geo location to the existing Bank_details table with data type varchar and size 20.

/* Solution */
Alter table Bank details add geo location varchar (20);

Ques 6) What is the Value of Geo location for product "PayCard"?

/* Solution */
Select geo location from Bank details where Product = 'PayCard'.

Output: NULL.

Ques 7) How many characters does the Product "paycard" have in the Bank details table.
/* Solution */

Select char_length (Product) from Bank_details where Product = 'PayCard'.

Output.
7

char_length (length of character)

Scanned with CamScanner

DATE / /
PAGE NO.

Question 8)
Alter the Product field from Char to Varchar in Bank_details

/* Solution */
Alter table bank_details modify PRODUCT Varchar(

Ques 9)
Reduce the size of the Product field from 10 to 6 and check if it is possible.

/* Solution */
Alter table bank_details modify product Varchar(6);

-Output
Error##

Ques 10)
Create a table named as Bank_Holidays with below fields
a) Holiday field which displays only date
b) Start time field which displays hours and minutes
c) End time field which also displays hours and minutes and timezone.

Scanned with CamScanner

Date / /
Page No.

/* Solution */

Create table bank holidays (
Holiday date;
start_time datetime,
End_time timestamp);

11)
11) Step 1 : Insert today's date details in all fields of Bank Holidays
Step 2 : After Step 1, perform the below
- Postpone Holiday to next day by updating the Holiday field.

/* Solution */
- Step 1 :-
Insert into bank_holidays Values (Current_date(), Current_date(), Current_date());

Step 2
Update bank_holidays
Set holiday = ~~Date~~ DATE ADD(Holiday, INTERVAL 1 DAY);

12) Update the End time with Current european time
/* Solution */

Update Bank Holidays Set End time = utc timestamp();

Scanned with CamScanner

DATE / /
PAGE NO.

Ques 13)
Display output of Product field as New-Product in Bank details table.

/* Solution */
Select Product as New-product from bank-details;

Ques 14)
Display only one record from bank details.
/* Solution */

Select * from bank_details limit 1;

Ques 15) Display the first five characters of the Geo-location field of bank_details.
/* Solution */

Select substr(Geo_location, 1, 5) from 'bank_details'

Done

Scanned with CamScanner

Date / /
Page No.

Additional

- Print details of employee whose first_name ends with 'A' and contains 6 alphabets.

Select * from Employees where first_name Like '_____A'.

Numeric Function.

- ABS(-32) = 32 (Negative to positive).
- MOD(18,4) = 2 (gives remainder)

  4) 18 (4
  16
  2

- POWER(4,2). = 16
- SQRT(9) = 3
- Floor(25.75) = 25
- CEIL(5.7) = 6
- GREATEST(30,2,31) = 31
- LEAST(2,34,81) = 2
- TRUNC(54.6656542,2) = 54.66 , (1234.5678, -2) = 1200
- ROUND(34.7652, 2) = 34.77
- SIGN(255) = 1
- LOG(9,81) = 2

Scanned with CamScanner

Character functions :-

- CHR(97) = gives result as to ascii code = a
- ASCII ('G') = gives ascii no. as 71
- INITCAP ⇒ will capital the first letter of words.
  INITCAP (hello world) ⇒ Hello World.
- CONCAT ⇒ Combine two or more strings
  CONCAT('amar', 'sharma') ⇒ amarsharma
- LPAD ('prince', 10, '*') ⇒ ***prince.
- RPAD ('prince', 10, '*') ⇒ prince****
- LTRIM ('prince', 'pr') ⇒ ince
- RTRIM ('prince', 'e') ⇒ princ
- SUBSTR ('www.google.com', 5, 6) ⇒ google

  5

  Substr ('Hello 2', 6) → 2
- LENGTH ('prince') → 6
- REPLACE ('prince', 'r', 'i') → piince
- TRIM ('V' from 'Valun') → alun
  Trim (' Shop ') → Shop
- LOWER ('PRINCE') → prince
- instr ('Valun', 'a') → 2
  (gives location of second argument)

Single row

Date / /
Page No.

exple

Select S*2 from dual;
S-2, S+2, ...

248.548387

Date Functions

i) Sysdate :- Select Sysdate from dual;

ii) Months_between :- Select months_between ('11-Mar-2021', '25-JUN-2000') from dual;

iii) Last_day :- Select LAST_day ("11-Mar-2021') from dual; → 31-Mar-21

iv) Next_day = Select next_day ("11-Mar-2021', 'Monday') from dual → 15-mar-21

v) Add_months = Select add_months ('11-Mar-2021', 5) from dual; → 11-AUG-21

vi) To-char = Select To-char (SYSDATE, 'yyyy-mm-dd') from dual; → 2021-03-11

vii) To-Date :- Select to_date ("11-03-21', 'dd-mm-yy' from dual; → 11-MAR-21

viii) Year

ix) TRUNC :- Select TRUNC(Sysdate) from dual; → 11-Mar-21

x) ROUND :- Select Round (Sysdate) from dual; 12-Mar-21

Scanned with CamScanner

DATE / /
PAGE NO.

General Functions.

- NVL
- UID :- Select UID from dual; — 9
- USER :- Select User from dual; — CS72221
- NVL2
- NULLIF
- COALESCE

Aggregate :-

- Count
- Avg
- Sum.

Relational Algebra

↳ we can't Apply it on any Computer

Scanned with CamScanner

DCL Commands.

- Grant → क्या Permission देनी है, कब लेनी है।
- Revoke → Permission हटाती है।

किसी Data पर से किसी User को Permission देनी है या लेनी है, तब इसे Use karte hain.

Mainly for Administrators.
(Developers)

- Roll ⇒ Permissions
(Set of permissions)

- Lab

Table XXSD.DEPT

DEPTNO	DNAME	LOC
10	Accounting	New York
20	Research	Dallas
30	Sales	Chicago
40	Operations	Boston

Other User want to Use this Table.
We will give it grant

Scanned with CamScanner

* Owner Table Should need to make Synonym for that particular Environment, Server or System.

Syntax :- Create Synonym XXSD.XXSD_DEPT for APPS.XXSD_DEPT

Drop ⇒ Drop Synonym XXSD.XXSD_DEPT
Describe = desc XXSD_DEPT

To give grant → My Username = Apps
Other user Username = XXSD

= Grant Select on XXSD_DEPT to XXSD
Table name
user id.

Now grant is Generated.

But.... It will only give permission to 'SELECT' from table.

= Grant INSERT on XXSD_DEPT TO XXSD

= Grant DELETE on XXSD_DEPT To XXSD

Scanned with CamScanner

To grant all permissions to a User:-

Date / /
Page No.

- GRANT SELECT, Insert, Update, Delete on XXSD_DEPT TO XXSD
- GRANT ALL ON XXSD_DEPT TO XXSD

☆ Revoke

- Revoke Select on XX SD_DEPT to XXSD.
- Revoke Insert - - - .
- - - - Delete - - - .

- Revoke Select, - - - - - .
- Revoke All on - - - - - - .
- Revoke all privileges on XXSD_DEPT - - - .

Grant:-

- Some other types of privileges:

a) Reference Privileges: To permit to declare foreign key while creating relations.

- Allow a user U_1 To create relation that refer to key 'Eid' of Emp-Salary relation.

Grant References (Eid) on Emp-Salary To U1;

Scanned with CamScanner

DATE / /
PAGE NO.

b) Execute Privileges :- Authorizes a user to execute a function or procedure.

- Allow a user U1 to execute privelege on a function Create_Acc() on call function.

Grant Execute on Create_acc() to U1;

Restrict

Restrict → ए जो कहा है वो ही वापस लेगा.
Cascade :- Saare Privelegs वापस ले लेगा।

TCL

Date / /
Page No.

~~For example~~

- Commit
- Rollback
- Savepoint

DEPTNO	Dname	Loc
10	Accounting	New york
20	Research	Dallas
30	Sales	chicago
40	operations	Boston

Commit

Select * from XXSD_DEPT
insert into XXSD_DEPT VALUES(1, 'OPMS1', 'Phone'1')
insert . . . . . (2, '' 2 '' 2)
insert . . . . . (3, '' 3 , '' 3)

~~User 1~~ Administrator

~~At~~ User 1
Select * from XXSD_DEPT
→ It will only show the records. Not new updated for User 1

If Administrator will ~~Enter~~ Enter Commit;
After ~~an~~ inserting record, then it will show the new data to User 1.

(It will save the data in database not in Cache memory)

Scanned with CamScanner

Rollback :-

Insert 3 More rows.

Total records = 9
and now I realized that I inserted wrong rows here After COMMIT को Execute करके

Then type Rollback;
The Data entered after Commit will be flushed

Savepoint

Enter data ;
Savepoint Stage1 ;
Enter data 2 ;
Savepoint Stage 2 ;

No Commit

Agar Sabhi सही है, तो Commit करें, सब Save ho Jayega.

~~Agar kuch galat lage~~
Agar aapko लगे सब कुछ galat hai, To Simple Rollback करें।

Agar Kuch galat hai, और Stage 4 ~~galat hai~~ और उसके बाद Data Entered galat hai.

Scanned with CamScanner

Date / /
Page No.

Then Simple Type.

- Rollback to Stage 3;

Functions (Confusing)

- Group by
- Having

Ex:- 1 Room hai, बहुत सारी सब्जीयाँ हैं → Sabza Dekh Separate karo, Unique karo, कितनी quantity hai.

Ex:- अगर किसी भी ढेर का Count 20 से कम है, उसको मत बताना।

Scanned with CamScanner

DATE / /
PAGE NO.

	Fruit_name	Unit_price	Quantity
1	Apple	100	60
2	Orange	110	40
3	Apple	105	45
4	Kiwi	190	22
5	Banana	70	20
6	Pumpkin	14	120
7	Banana	60	23
8	Tomato	13	15
9	Pumpkin	11	12
10	Tomato	22	100
11	Apple	101	11
12	Kiwi	200	19
13	Apple	115	12
14	Tomato	11	21
15	Banana	65	25
16	Pumpkin	16	33
17	Apple	105	12
18	Tomato	12	29
19	Orange	135	21
20	Kiwi	195	26

Group by

Apple	140
Banana	68
Kiwi	67
Orange	61
Pumpkin	165
Tomato	165
Grand total	666

Scanned with CamScanner

(Having = where)

Having

DATE / /
PAGE No.

Apple	140
Tomato	165
Grand Total	666
pumpkin	165

i) Select fruit_name, Count(*) QUANTITY from XXSD_FOOD_TL Group by fruit_name

ii) Select fruit_name, Sum(Quantity) Quantity from XXSD_FOOD_TL Group by fruit_name

Having :- Same ii)
Having Sum(Quantity) > 100

(Having Aggregates par lagti hai — Sum, Max, Min - - - -)

Where को Use kar सकते हैं, लेकिन Group by के पहले :-

Select fruit_name, Sum(Quantity) Quantity
from XXSD_Food_TL
where fruit_name in ("Apple", "Pumpkin")
Group by fruit_name
Having Sum(Quantity) > 100

Scanned with CamScanner

Date / /
Page No.

Group by:

The group by clause is used ~~to~~ in Conjunction with the Select statement and Aggregate functions to group rows together by common column values.

Syntax:-
```
Select Columns
from table_name
where Condition
Group by Column_name(s);
```

Syntax:-
```
Select Columns
from Tabe1 INNERJOIN Table2
ON Table1.Column_name = table2.column_name
where Condition
Group by Column_name(s);
```

Scanned with CamScanner

MySQL DATE TIME Data types :

To Store date or date/time value in the database :

- DATE - format : yyyy-mm-DD
- Time : HH : MI : SS
- Datetime - format : yyyy-mm-dd HH : MI : SS

```
Create table hosp (
    id int not null primary key,
    fname varchar2(20),
    dob date,
    dotime time,
    dat datetime);
```

Extra :-
auto-increment

=) Values (1, 'Abhi', '1994-01-05', '05:05:SS', '1994-01-05 05:05:SS');

Predefined functions :-

CURDATE() → Give the Current date : YYYY-mm-DD
NOW() :- Gives both Cur date & Time : YYYY-mm-DD HH:MI:SS
CURTIME() → Gives Current time : HH : MI : SS

Values('thapa', curdate(), curtime(), now());

Scanned with CamScanner

Date / /
Page No.

☆ Select * from childdata

बच्चे कब 18 के होंगे?

क्या (निकालना है) Columnname

Select *, DATEADD(YEAR, 18, DateofBirth) as Validdob from childdata.

कितने साल बाद

Datediff.

→ DD
→ month

Select datediff(Year, '2011-02-20', '2015-06-25')

Hour → (hh
minute → (mi)

आज की Date में पिछले 4 महीने से कोई order नहीं किया है?

Select * from CustomerOrderdata where Datediff (Month, Orderdate, Getdate()) >= 4

Month

Monthname()

Select * Monthname('2020-11-04');

Day :- Select day('2020-11-06');

Dayname

Day of month() :- Returns the day of the given date

Day of week() :- Returns the number of week day (1 - Mon-day, 2 - Tuesday ...)

Day of year() : Day of the given date.

(1-31) Jan
1 feb = 32 Days no. of year.

Scanned with CamScanner

te ← DDL

DML → Delete

k

oe Bina Condition लगाए भी Data Delete kar sakte hain.
र Condition Lagana padti hai.

Some Queries.

Select * from emp where ename like('%R') and 'Comm' is NULL;

Select ename, empno from emp where job = 'clerk' order by empno;

Select Ename, hire_date, deptno from emp where to_char(hiredate, 'yyyy') > 1975 and deptno = 20;

Scanned with CamScanner

Table Student

Date / /
Page No.

```
Create table Student
(
 Name Varchar2(20) Constraint name_null Not null,
 Roll_no Decimal Primary key,
 class Varchar2(20) Not Null,
 Dob Date Not Null,
 Marks Decimal(6) check( Marks between 0 and 100),
Section Char(6) check( Section in ('A','B','C','D')),
Grade char(5) check (Grade in ('A','B','C','D')),
check (( Marks>=80 and grade='A') or (Marks between 60 and
79 and grade='B') or( Marks between 40 and 59 and grade
='C') or( marks<40 and grade='D')),
Sex char(3) check( Sex in ('m','f')),
City Varchar2(40) Not null
)
```

- Alter table Student modify City Varchar2(20) default 'Delhi';

- Drop the not null Constraint from "name" Column.

→ Alter table Student drop

- List all students who Secured grade B and who were born before 1985.

→ Select * from Student where grade='B' and to_char (DOB,'yyyy')<'1985';

Scanned with CamScanner

Date / /
Page No.

- List all details of Students with even roll nos.

→ Select * from Student where mod(Roll-no, 2) = 0;

→ Select * from Student where to_char(DOB, 'DY') = 'SUN' and To_char(DOB, 'mon') = 'DEC' and to_char(DOB, 'yyyy') < 1990';

→ Select * from Student where City = 'Delhi' and name like 'K%' and Sex = 'M';

→ Update Student set Marks = Marks+5 where Marks <= 95 and City Like 'Delhi';

→ Delete from Student where city in('Delhi', 'Bombay') and marks > 50';

Scanned with CamScanner