SQL Cheat Sheet

1. DBMS

- Hierarchical DBMS
- Network DBMS
- + Relational DBMS
- Object-Relational DBMS (PostgreSQL)

Oracle SQL & PL SQL

RDBMS → Collection of programs to access Data.
Collection of Tables

Cardinality → Uniqueness of Data in [Columns] / No. of Tuples.
High Cardinality → More unique values in column.
Low Cardinality → low unique values in columns.

Table (Relation, File, Class)
Column (field, Attribute)
Row (Record Tuple)
Query → Command retrieving info from Server.

SID	SNAME	LOCATION
1	Peter	London
2	Michael	Paris
3	John	Mumbai
4	Henry	London.

Relation (SID, SNAME, LOCATION) Cardinality → 4 Degree - 3

Degree → Uniqueness of Data in [Tuples.] / No. of columns
Attribute → Property Describing an entity
Domain → Set of Permitted values (A-Z, a-z, " ") SNAME
(0-9) SID

Data Integrity → Accuracy
Consistency
Constraints.

Entity Integrity (PRIMARY)
Domain Integrity (DOMAIN)
Referential Integrity (FOREIGN)

Keys →

Primary keys - Unique
Alternate key → All the unique keys except the primary keys.
Candidate keys - Set of Attributes, to be unique.

SQL Cheat Sheet

Foreign key – Cross Reference between 2 Tables referencing primary key of other Table

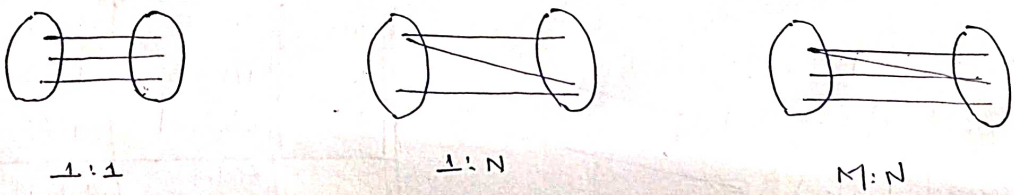
CAN BE MORE THAN 2

Entity Relationship model (E.R.)

Entity → Employee, Computers

Attribute → Name, Salary

Cardinality of Relationships –

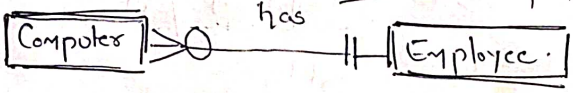


1 : 1 1 : N M : N

Crow foot Notation

Exactly One One or Zero Zero, One, More

Entities →



SQL Basics.

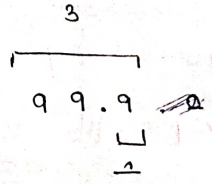
CHAR	VARCHAR
String.	Alphanumeric
255 Charac.	65355
fixed	Variable
Static Memory All.	Dynamic Memory Allocation.

INTEGER, SMALL INT, INT

NUMERIC, DECIMAL, NUMBER (3,1)

Precision

Scale.



CHAR, VARCHAR, CLOB, BLOB

SQL Cheat Sheet

② DATE DD-MON-YY 02-JUL-81 ILP(99)
TIMESTAMP 20-JUL-69 08.18.00.000000 PM

LOGICAL OPERATORS -

① RANGE Salary BETWEEN 25000 AND 30000
② LIST Dept IN ('IVS', 'ETA', 'ICP')
③ LIKE SuppliedId LIKE 'S%'
④ NULL BOSS IS NULL

AND — if 1 & 2 are true
OR — if any 1 is true
NOT — ID NOT IN (2,3)

CREATE

```
CREATE TABLE STUDENT ( StudentId INTEGER,
                       GENDER CHAR(1),
                       FNAME VARCHAR(10),
                       DOJ DATE);
```

DROP TABLE STUDENT;

Specify Rules
Constraints
- NOT NULL (Column level)
- PRIMARY KEY
- UNIQUE
- CHECK
- FOREIGN KEY
- DEFAULT

→ StudentID INTEGER CONSTRAINT Stud_Sid_pk PRIMARY KEY,
CONSTRAINT Stud_SID_nn NOT NULL,
DOJ DATE DEFAULT SYSDATE;
Gender CHAR(1) CONSTRAINT Stud_Gender_chk CHECK(GENDER IN('M','F')),

Note → After Declaring all column if we declare constraint
— TABLE LEVEL CONSTRAINT
NOT NULL IS COLUMN LEVEL CONSTRAINT

Ex→ CONSTRAINT Stud_Cno_uk UNIQUE (ContactNo.));
CONSTRAINT Marks_sid_fk FOREIGN KEY (StudentID) REFERENCES Student(StudentID);
OR — TABLE LEVEL
CONSTRAINT Marks_sid_fk REFERENCES Student(StudentID), — Column level

INTEGRITY CONSTRAINTS used to maintain the quality of information such that insertion, updating, etc.. to be performed without compromising data integrity

Ex→ Column check constraint, cannot check another columns.

FNAME VARCHAR(10)
LNAME VARCHAR(10) CHECK (FNAME <> LNAME); ✗
, CHECK (FNAME <> LNAME) ✓

SQL Cheat Sheet

Domain Cost.	Entity Integrity Constraint.	Referential Integrity Constraint	Key Constraint
Any attribute must be 3000	PRIMARY KEY NOT NULL	FOREIGN KEY B/W 2 TABLE	LIKE PRIMARY KEY MUST BE UNIQUE

PRIMARY	NOT NULL	(only 1)
UNIQUE	NULLABLE	(more than 1)

ALTER

To change structure of existing table, Add'n of column
datatype change
add remove constraints.

ALTER TABLE STUDENT ADD Address VARCHAR2(20)
MODIFY Address Varchar 2(50)
RENAME COLUMN Address TO ResAdd.
DROP (ResAdd)
ADD CONSTRAINT studID_pk PRIMERY KEY (STUDID)
DROP CONSTRAINT studID-pk

DAY 3.

INSERT RETRIEVE UPDATE DELETE] DML

INSERT

① Insert into EMPLOYEE
VALUES (1, "Jame", 4000, NULL, 1)

OR

② INSERT into EMPLOYEE (Id, Ename, Salary, Bonus, EmpID)
VALUES (1, "Jame", 40000, NULL, 1)

OR

③ INSERT into EMPLOYEE (Id, Ename, Salary, Bonus, EmpID)
SELECT QUERY

SELECT

SELECT IDNAME, ENAME, SALARY FROM EMPLOYEE;

SELECT IDNAME AS EmpID, Ename AS EmpName FROM EMPLOYEE
WHERE SALARY < 40000

SELECT ENAME, 30 AS VALUE from EMPLOYEE Constant

SELECT SALARY * 2 AS DOUBLE-SALARY from EMPLOYEE Expression

Duplicates.

Select DISTINCT Dept, MANAGER FROM EMPLOYEE;

SQL Cheat Sheet

(3) To select clause do not remove duplicates hence DISTINCT (Performance issues)

WHERE

SELECT ID, Ename from EMPLOYEE WHERE SALARY >= 30000 AND DEPT = 'ETA';

① BETWEEN 30000 AND 40000 — BONUS = NULL / IS NULL — AND OR >< = BETWEEN

WHERE ID IN (2,3) — BONUS IN NULL

WHERE ID NOT IN (2,3)

② WHERE Designation = 'PM' / 'pm'

Ename = 'James Potter'

} Trailing Spaces ignored in CHAR
Leading spaces not ignored in CHAR
VARCHAR (Not ignored)

LIKE

SELECT ID, ENAME from EMPLOYEE where ENAME LIKE 'E%';

Start & End Pattern — %E , & %.

Anywhere — %m%.

FIXED — ' - - ,

→ '_a%' Second CHAR Starting with a.

F J W G H S D O

From Join where Group by Having Select Distinct Order By

UPDATE (The Row NOT DB Structure)

UPDATE Employee SET SALARY = SALARY * 1.2 , BONUS = 200 | WHERE ID = 5;

BONUS = SALARY * 0.40%

DELETE (Deleting Rows & Records, not Structure)

→ DELETE from Employee WHERE Dept = 'ETA'

→ TRUNCATE TABLE Employee, [Deletes every Row]

Error Codes.

ORA-00000 Successful Completion

ORA-00001 Unique Constraint Violation

ORA-00904 Invalid Identifier (Invalid Column or missing Column)

ORA-0913 Too Many Values More / Multiple values passed at once.

SQL Cheat Sheet

SQL functions. (Data Manipulations)

Single Row ✗ Multi Row functions.

Numeric functions -

```
Select City, MinTemp, CEIL(MinTemp) AS "Ceiling",     CEIL   High
       FLOOR(MinTemp) AS "Floor"                      FLOOR  low
       ABS(MinTemp) AS "Absolute"                     ABS    -to +
       ROUND(MinTemp, 1) AS "Round"                   ROUND
  From Weather;
```

Character Functions -

```
Select City, LENGTH(City) "CityLength",     LENGTH
       Lower(city) "Lowercase",             LOWER
       UPPER(City) "Uppercase",             UPPER
       CONCAT('City is', city),             CONCAT, ||, Nested Concat
       City || Country "Concat by Operator", from WEATHER;
       SUBSTR(City, 2, 10) "Ten_from_2"     ] SUBSTR
       SUBSTR(City, 3), SUBSTR(City, 7, 2)  ] (Val, from, N)
```

Conversion function -

```
TO_CHAR(1210.73, '9,999.99')        TO_CHAR(Sysdate, 'FM Mon DDth, YYYY')
   1,210.73                            JUL 9TH, 2003        FM 09 8 - 5pp.

TO_DATE('20020315', 'YYYYMMDD')
   date value of Mar 15, 2002

TO_NUMBER('23', '99')
```

TO_CHAR
TO_DATE
TO_NUMBER

To_Char can be used to format, extract dates.

Eg → TO_DATE(Record Date, 'Mon')

Date functions.

```
Select SYSDATE, SYSTIMESTAMP from DUAL;
ADD_MONTHS('01-Aug-03', -3)
           ('31-Jan-03', 1)

MONTHS_BETWEEN(TO_DATE('2003/01/01', 'YYYY/MM/DD'),   -2.41935483
               TO_DATE('2003/03/04', 'YYYY/MM/DD'))

MONTHS_BETWEEN('01-Feb-2014', '01-Jan-2014') "MonthsDif"   1
```

SYSDATE
SYSTIMESTAMP
ADD_MONTHS
MONTHS_BETWEEN

SQL Cheat Sheet

(4) Aggregate functions -

Select MIN(Salary), MAX(Salary), SUM(Salary), AVG(Salary)

Select Count(*) from table_name;

Count (DISTINCT emp_name)

10000 MIN COUNT
50000 MAX SQRT
4,50000 SUM
28333.3 AVG

Count Does not counts 'NULL'

DISTINCT only on Column

MIN, MAX apply on Character in LexicoGraphic Order.

Misc functions.

NVL (value1, value2) Subs val.1 by val2 if found NULL

NVL (City, "Not_Available") NVL & USER

Select USER from DUAL;

CASE

Simple Case Searched Case.

```
UPDATE Student set Result =
        CASE
        WHEN Mark >=300 THEN 'First'
        WHEN Mark<300 AND Mark >=250 THEN 'SECOND'
        WHEN MArk < 250 AND Mark >=150 THEN 'THIRD'
        ELSE 'FAIL'
        END;
from Student;
```

Searched Case.

```
CASE Designation
        WHEN 'SE' THEN Salary * 1.2
        WHEN 'SSE' THEN Salary * 1.1
        ELSE Salary *1
        END New_SALARY
```

Simple Case.

SQL Cheat Sheet

Group By Order By. Asc | DESC

Ascending Order By default

```
Select * From Customers ORDER By Country DESC;
SELECT * FROM Customers ORDER BY Country ASC, CustomerName DESC;
```

SQL Wild Cards.

%	_	[ ]	^	-
bl %	h_t	h[oa]t		c[a-b]t
black	hot	hat	hit	cat

As → AliasName

```
Select CustomerName, Address + ', ' + PostalCode + ', ' + City + ', ' +
       Country As Address.
```

Abhinendra Sharma.	1/109, 208002, Kanpur, India
CustomerName.	Address

Changing Table Names Also.

```
Select O.OrderID, O.OrderDate, CCustomerName
  from Customers As C, Orders As O,
Where C.CustName = 'Arnold' AND C.CustID = O.OrderID;
```

JOINS.

INNER JOIN | LEFT JOIN | RIGHT JOIN | FULL OUTER JOIN

CROSSJOIN | CARTESIAN PRODUCT

```
Select * from Employee CROSSJOIN Department;
Select * from Employee, Department;
```

INNER JOIN

```
SELECT Column_name FROM table1 INNERJOIN table2
    ON column_name = columnname.        EQUI
       column_name <> column_name       NON-EQUI

SELECT column_name FROM table1 NATURAL JOIN table2
    (Primary key name = Foreign key name)
```

SQL Cheat Sheet

Left OUTER JOIN / LEFT JOIN

The left join keyword returns all rows from left table with the matching rows in right table.

If NULL match in right table, then no match

Example →

Left Join → If we want to append info's about orders to customers regardless whether a customer has placed order or not.

	Name	Last_Name	Order_Date	Order_amt.
1.	George.	Shelby	07/14/1776	234.36
2.	Alex	Jefferson.	NULL	NULL
3.	Alicia	Denver.	08/12/1996	65.50

RIGHT OUTER JOIN

Perspective from table 2.

Syntax.

```
Select  Table1.Column1 , Table1.Column2 , Table2.Column1 , . . . .
FROM    Table1  [LEFT JOIN | RIGHT JOIN]  Table2
ON      Table1.MatchingColumn = Table2.MatchingColumn.
```

When we want No. of Columns to be less. we apply Natural Join.

Hash Joins. → Build Phase (In Memory hash index (Left Side of input))
→ Probe Phase.

Large Table or Instance with Index

Self Join

To apply Join on 3 Table, 2 Joins are used.

SQL Cheat Sheet

TRIGGER → Event Driven Actions.

Stored Procedure → Algorithm (No Return)

Group by vs Partition by

Group By	Partition By
Rows collapse into group.	Rows does not collapse
Rolling up rows and giving aggregate functions - AVG in groups	Same Rows and aggregate function : AVG in each rows

CTE - (Similar to SubQuery)

Since not stored in any temp file

```
WITH CTE-Emp as
   (

   )

Select * from CTE-Employee.
```

Temp Table →

EmployeeDemographics

- Insert Data from Bigger Table into Temp Table for Analysis

Stored Procedures →

- Algorithms running again
- If there is a temp table in STORED PROCEDURE

  PUT [DROP TABLE IF EXISTS #Temp Employee ;]