SQL

Introduction

- Databaseiscollectionofdata•
  StructuredQueryLanguage(SQL)letsyouperformCRUDoperationsCreate/Read/UpdateorDeleteoperationsonadatabase

•DatabaseManagementSystem(DBMS)isasoftwareinterfacebetweendatabaseandenduserthatisresponsibleforauthentication, concurrency, logging, backup, optimization etc.

•Therearemanytypesofdatabase-Relational,Hierarchical,Network,NoSQLetc. •

Examples of RDBMS

- ∘MySql – OpenSource
- ∘SQLServer–Microsoft ∘
- Oracle – IBM
- ∘PostgreSQL - OpenSource

FollowingtutorialisforSQLwithMySqldbms.Thesamecanbeusedforotherdbmswithlittleornomodifications. Link to

install Mysql - https://dev.mysql.com/downloads/

DataTypes

CharacterData

char-eg:char(5)storesfixedlengthstringoflength5.Max255bytes.

varchar-eg:varchar(5)storesvariablelengthstringoflength5.Max65535bytes.

```
SHOWCHARACTERSET;--showsvariouscharactersetsthataresupported.
```

latin1isthedefaultcharacterset.Wecanalsochooseaspecificcharactersetlikebelow,

```
varchar(10)charactersetutf8       --theparticularcolumnissettoutf8
createdatabaseforeign_salescharactersetutf8;        --entiredatabaseissettoutf8
```

TextData

AllimagesinthesectionarefromLearningSQLbyAlanBeaulieu

Text type	Maximum number of bytes
Tinytext	255
Text	65,535
Mediumtext	16,777,215
Longtext	4,294,967,295

BLOB-BinaryLargeObjectFile→TinyBlob,Blob,MediumBlob,LongBlob

NumericalData

WholeNumbers:

Type	Signed range	Unsigned range
Tinyint	−128 to 127	0 to 255
Smallint	−32,768 to 32,767	0 to 65,535
Mediumint	−8,388,608 to 8,388,607	0 to 16,777,215
Int	−2,147,483,648 to 2,147,483,647	0 to 4,294,967,295
Bigint	−9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	0 to 18,446,744,073,709,551,615

TinyintisusedtostoreBoolorBoolean

DecimalNumbers:

Y

Type	Numeric range
Float(*p*,*s*)	−3.402823466E+38 to −1.175494351E-38 and 1.175494351E-38 to 3.402823466E+38
Double(*p*,*s*)	−1.7976931348623157E+308 to −2.2250738585072014E-308 and 2.2250738585072014E-308 to 1.7976931348623157E+308

TemporalData

Type	Default format	Allowable values
Date	YYYY-MM-DD	1000-01-01 to 9999-12-31
Datetime	YYYY-MM-DD HH:MI:SS	1000-01-01 00:00:00 to 9999-12-31 23:59:59
Timestamp	YYYY-MM-DD HH:MI:SS	1970-01-01 00:00:00 to 2037-12-31 23:59:59
Year	YYYY	1901 to 2155
Time	HHH:MI:SS	-838:59:59 to 838:59:59

BasicQueries

Databasequeries

```
CREATEDATABASELOGICFIRST;--createsanewdatabase
-- TO DELETE A DATABASE
DROPDATABASELOGICFIRST;
DROPSCHEMALOGICFIRST;--sameasabove.ucanuseDATABASEOrSCHEMA
DROPSCHEMAIFEXISTSLOGICFIRST;--preventserrorifdbnotfound

SHOWDATABASES;--showsallthedatabases
SHOWSCHEMAS;--sameasabove.showsschemas/db

USE SYS; -- uses this database for all further commands
SHOWTABLES;--showsalltablesinthedatabasebeingused
```

Table-Create,Delete,Alter

primarykey-uniquelyidentifiesarowinatable

```
//creating  a  table
CREATETABLEstudent(
  idINTPRIMARYKEY,
    nameVARCHAR(30),
    gpa DECIMAL(3,2)
);
--·-----or------
```

```
CREATETABLEstudent(
  id INT,
    nameVARCHAR(30),
    gpaDECIMAL(3,2),
    PRIMARY KEY(id)
);

DROPTABLEstudent;--dropsthetable

DESCRIBEstudent;--describesthecolumnsinthetablestudent

ALTERTABLEstudentADDdepartmentVARCHAR(5);--Addsanewcolumndepartmenttothestudenttable

ALTERTABLEstudentDROPCOLUMNdepartment;--dropsthedepartmentcolumnfromstudenttable
-----or---
ALTERTABLEstudentDROPdepartment;--sameasabove
```

Inserting Data

```
INSERTINTOstudentVALUES(1,"Aarthi",7.6);
INSERTINTOstudentVALUES(2,"Anitha",8.5);--insertsarow.givevaluesincolumnorder

INSERTINTOstudentVALUES
(3,"Anitha",8.5),
(4,"Arul",8.2),
(5,"Ashwin",7.6);--insertsmorethanonerow

INSERTINTOstudent(id,name)VALUES(5,"Balaji"),(6,"Chandru");--insertsspecificcolumns.
```

Select

```
SELECT*FROMstudent;--displaysallrowsandcolumnsinthestudenttable SELECT id,name
FROM student; -- displays specific columns
```

WhereClauseandConditions

whereisusedtofiltertherecords.Therowsarefilteredbasedonconditions.

▼**QueryforEmployeetable**(clicktheinitialarrowtoexpand)

```
CREATETABLEemployee(
emp_idINTPRIMARYKEY,
ename VARCHAR(30),
job_descVARCHAR(20),
salary INT );

INSERT INTO employee VALUES(1,'Ram','ADMIN',1000000);
INSERT INTO employee VALUES(2,'Harini','MANAGER',2500000);
INSERT INTO employee VALUES(3,'George','SALES',2000000);
INSERT INTO employee VALUES(4,'Ramya','SALES',1300000);
INSERT INTO employee VALUES(5,'Meena','HR',2000000);INSERT
INTO employee VALUES(6,'Ashok','MANAGER',3000000); INSERT
INTO employee VALUES(7,'Abdul','HR',2000000);INSERT INTO
employee VALUES(8,'Ramya','ENGINEER',1000000); INSERT INTO
employee VALUES(9,'Raghu','CEO',8000000);
INSERTINTOemployeeVALUES(10,'Arvind','MANAGER',2800000);
INSERTINTOemployeeVALUES(11,'Akshay','ENGINEER',1000000);
INSERT INTO employee VALUES(12,'John','ADMIN',2200000);
INSERTINTOemployeeVALUES(13,'Abinaya','ENGINEER',2100000);
```

```
SELECTcolumn1,column2,...
FROM table_name
WHEREcondition;
```

Followingcanbeusedwithinthecondition.

=	Equal
>	Greaterthan
<	Lessthan
>=	Greaterthanorequal
<=	Lessthanorequal
<>	Notequal.Note:InsomeversionsofSQLthisoperatormaybewrittenas!=
BETWEEN	Betweenacertainrange
LIKE	Searchforapattern
IN	Tospecifymultiplepossiblevaluesforacolumn
NOT	negation

AND/ORcanbeusedtocombinetherelationaloperators.

```
SELECT*FROMemployee
WHEREename='Ramya';

SELECTemp_id,ename,salaryFROMemployee
WHERE salary>2000000;

SELECT emp_id,ename,salary FROM
employeeWHEREsalary<2600000ANDjob_desc='MANAG
ER';

SELECT*FROMemployee
WHEREjob_desc='ADMIN'ORjob_desc='HR';--thoughthisworksnextcommandisamuchbetterchoice

SELECT*FROMemployee
WHEREjob_descIN('ADMIN','HR');

SELECT*FROMemployee
WHEREjob_descNOTIN('MANAGER','CEO');

SELECT*FROMemployee
WHEREsalaryBETWEEN2000000AND3000000
LIMIT5;--limitstherecordsshown

SELECT*FROMemployee
LIMIT5;--Differentsyntaxinoracle/sqlserver
```

Using Like and wildcards

LIKEisusedwithWHEREclauseforsearchingaspecificpatterninacolumn.Itisusedalongwiththefollowingwildcards

%representszeroormorecharacters

_representsexactlyonecharacter

```
SELECT*FROMemployee
WHEREenameLIKE'A%';--filtersnamestartingwithA

SELECT*FROMemployee
WHEREenameLIKE'R%a';--filtersnamestartingWITHRANDendingwitha

SELECT*FROMemployee
WHEREenameLIKE'%I%';--filtersnamecontainingI
```

```
SELECT*FROMemployee
WHEREenameLIKE'I%';--filtersnamewithiasthirdcharacter

SELECT*FROMemployee
WHEREenameLIKE'R\%';--filtersnamestartingwithR%.\istheescapecharacter.
```

UpdateandDelete

```
UPDATEemployee
SETjob_desc="Analyst";--updatesalljob_descofallrowsto"Analyst"whensafeupdatenotenabled

UPDATEemployee
SETjob_desc="Analyst"
WHEREjob_desc="Engineer";--changesEngineertoAnalystinallapplicablerows

UPDATEemployee
SETjob_desc="Analyst"
WHERE emp_id=1;

DELETEFROMemployee;--deletesallrows

DELETEfromemployee
WHERE emp_id = 12;
```

Distinct

```
SELECTDISTINCTjob_desc
FROMemployee;--showsonlydistinctvalueswithoutduplicates
```

OrderBy

```
SELECT*FROMemployee
ORDERBYsalary;--orderbysalaryasc

SELECT*FROMemployee
ORDERBYsalaryASC;--orderbysalaryasc

SELECT*FROMemployee
ORDERBYsalaryDESC;--orderbysalarydesc

SELECT * FROM employee
WHEREjob_desc="MANAGER"
ORDERBYsalaryDESC;--orderthemanagersalariesindescorder

SELECT*FROMemployee
ORDERBYjob_desc,ename;--firstsortsbyjob_descandthenbyename

SELECT*FROMemployee
ORDERBY(CASEjob_desc--specificorder WHEN
'CEO' THEN 1
WHEN'MANAGER'THEN2
WHEN'HR'THEN3
WHEN'ANALYST'THEN4
WHEN 'SALES' THEN 5
ELSE 100 END);
```

CopyTable

```
INSERTINTOfirst_table_name[(column1,column2,...columnN)]
   SELECT column1, column2, ...columnN
   FROMsecond_table_name
```

Functions

- Hereisgoodsourceforlearningallfunctionshttps://www.techonthenet.com/mysql/functions/index.php aggregate functions https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions.html

```
SELECTCOUNT(*)FROMemployee;--totalcountofentriesinthetable SELECT

AVG(salary) FROM employee; -- avg salary of all employees SELECT

AVG(salary)
FROMemployee
WHEREjob_desc="MANAGER";--avgsalaryofmanagers

SELECTSUM(salary)
FROM employee
WHEREjob_desc="ANALYST";--totalsalarygiventoallanalysts

SELECT*FROMemployee
WHEREsalary=(SELECTMAX(salary)
FROMemployee);--displaytheemployeewith SELECT

MIN(salary) FROM employee;

SELECT UCASE(ename),salary
FROMemployee;--uppercase

SELECT ename,CHAR_LENGTH(ename)
FROM employee;

SELECTename,CONCAT("Rs.",salary)
FROMemployee;--addsRs.tothebeginningofsalary

SELECTename,CONCAT("Rs.",FORMAT(salary,0))
FROMemployee;--formatsthenumbertoaddcomma.Thesecondargument(0here)representsdigitstoroundoffafterdecimal

SELECTename,LEFT(job_desc,4)
FROMemployee;--returnsonlythefirst4charactersoftheename
```

UsingDate

```
ALTERTABLEemployeeADDCOLUMNHire_DateDATE;--addinghire_datecolumn


UPDATEemployee
SETHire_Date="2012-10-05";--updatinghire_date

UPDATEemployee
SETHire_Date="2014-10-05"
WHEREjob_desc="ANALYST";--updatinghire_date SELECT

NOW(); -- Current date and time

SELECT DATE(NOW()); -- current date

SELECTCURDATE();--currentsystemdate

SELECTDATE_FORMAT(CURDATE(),'%d/%m/%Y');--tochangethedisplayformat.use%d%mand%yor%Yinrequiredformat.

SELECTDATEDIFF(CURDATE(),'2020-01-01')DAYS;--tocalculatedatedifference
```

```
SELECT CURDATE() 'start date',
 DATE_ADD(CURDATE(),INTERVAL 1 DAY) 'one day later',
 DATE_ADD(CURDATE(),INTERVAL 1 WEEK) 'one week later',
 DATE_ADD(CURDATE(),INTERVAL1MONTH)'onemonthlater',x
 DATE_ADD(CURDATE(),INTERVAL 1 YEAR) 'one year later';
```

start date	one day later	one week later	one month later	one year later
2022-02-12	2022-02-13	2022-02-19	2022-03-12	2023-02-12

GroupByandHaving

Groupbyisusedtogroupthetablebasedonconditionsandanalyzevalueswithinthegroupusingaggregatefunctions. Where is used to filter the rows before grouping. Having is used to filter the groups.

```
SELECTjob_desc,FORMAT(AVG(salary),0)avg_sal
FROM employee
GROUPBYjob_desc;--showsavgsalaryofeachcategorywithinjob_desc
```

job_desc	avg_sal
ADMIN	1,000,000
ANALYST	1,366,667
CEO	8,000,000
HR	2,000,000
MANAGER	2,766,667
SALES	1,650,000

```
SELECTjob_desc,COUNT(emp_id)count
FROM employee
GROUPBYjob_desc;--displaysnumberofemployeescountineachjob_desccategory
```

job_desc	count
ADMIN	1
MANAGER	3
SALES	2
HR	2
ANALYST	3
CEO	1

```
SELECTjob_desc,COUNT(emp_id)AScount--usingasforaliasing FROM
employee
GROUPBYjob_desc
HAVINGCOUNT(emp_id)>1;--displaysnumberofemployeescountineachjob_desccategoryonlywhencountisgreaterthan1.

SELECTjob_desc,COUNT(emp_id)AScount FROM
employee
GROUP BY job_desc
HAVINGCOUNT(emp_id)>1
ORDERBYjob_desc;--sameasaboveorderedbyjob_descasc

SELECTjob_desc,COUNT(emp_id)AScount FROM
employee
GROUP BY job_desc
HAVINGCOUNT(emp_id)>1
ORDERBYCOUNT(emp_id)DESC--samebutorderedbyDescorderofCOUNTineachgroup

SELECTjob_desc,COUNT(emp_id)AScount FROM
employee
WHERE salary>1500000
GROUP BY job_desc
HAVINGCOUNT(emp_id)>1
ORDERBYCOUNT(emp_id)DESC;--withadditionalfilteringofsalary>15L.Onlythosewithsalmorethan15Lisconsideredforgrouping
```

Constraints

NOTNULL,AUTO_INCREMENT,DEFAULT,CHECK,UNIQUE

```
CREATETABLEemployee(
emp_idINTPRIMARYKEYAUTO_INCREMENT,--idwillbeautoincrementedfornewrows
enameVARCHAR(30)NOTNULL,--nullvaluecannotbeinsertedforthecolumn
job_descVARCHAR(20)DEFAULT'unassigned',--setsdefaultwhennotmentioned salary INT,
panVARCHAR(10)UNIQUE,--cannotcontainduplicates
CHECK (salary>100000));

INSERTINTOemployee(ename,salary)VALUES('Ramya',1000000);
INSERTINTOemployee(ename,salary)VALUES('Riya',10000);--errosbecauseofviolationofsalarycheckconstraint SELECT * FROM
employee;
```

ForeignKey

Foreignkeyisafieldinonetablereferringtotheprimarykeyofanothertable.

```
--droppreviouslycreatedtablesandcreateabranchtable CREATE
TABLE branch (
branch_idINTPRIMARYKEYAUTO_INCREMENT,
br_name VARCHAR(30) NOT NULL,
addrVARCHAR(200));

--createemployeetablewithbranch_idasforeignkey.Itreferstothebranch_idofbranchtable. CREATE TABLE
employee (
emp_idINTPRIMARYKEY,
ename VARCHAR(30),
job_descVARCHAR(20),
salaryINT,branch_id
INT,
CONSTRAINTFK_branchIdFOREIGNKEY(branch_id)REFERENCESbranch(branch_id));

--dropingFK
ALTERTABLEemployee
DROPFOREIGNKEYFK_branchId;
```

Index

Indexareusedforfastlookups.Speedsupselectquerybutdelaysinsert/update.Alsotakeupmorememory.

```
SHOWINDEXFROMemployee;--showcurrentindices

CREATEINDEXname_indexONemployee(ename);--createsanewindex ALTER TABLE

employee
DROPINDEXname_index;--dropindex

ALTERTABLEemployee
ADDINDEX(ename);--createindexusingaltercommand
```

OnDelete

```
CREATETABLEemployee(
emp_idINTPRIMARYKEYAUTO_INCREMENT, ename
VARCHAR(30) NOT NULL,
job_descVARCHAR(20),
salaryINT,branch_id
INT,
```

```
CONSTRAINTFK_branchIdFOREIGNKEY(branch_id)REFERENCESbranch(branch_id)
ONDELETECASCADE--ondeletingarowinbranchtable,thecorrespondingentriesinemployeetablewillbedeleted
);

CREATETABLEemployee(
emp_idINTPRIMARYKEYAUTO_INCREMENT, ename
VARCHAR(30) NOT NULL,
job_descVARCHAR(20),
salaryINT,branch_id
INT,
CONSTRAINTFK_branchIdFOREIGNKEY(branch_id)REFERENCESbranch(branch_id)
ONDELETESETNULL--ondeletingarowinbranchtable,thebranchidcorrespondingentriesinemployeetablewillbemadenull
);
```

Joins

Joinsareusedtojoincolumnsfromtwotables

```
DROPTABLEemployee;--dropandfreshlycreate CREATE

TABLE branch (
branch_idINTPRIMARYKEYAUTO_INCREMENT,
br_name VARCHAR(30) NOT NULL,
addrVARCHAR(200));

CREATETABLEemployee(
emp_idINTPRIMARYKEYAUTO_INCREMENT, ename
VARCHAR(30) NOT NULL,
job_descVARCHAR(20),
salaryINT,branch_id
INT,
CONSTRAINTFK_branchIdFOREIGNKEY(branch_id)REFERENCESbranch(branch_id)
);

INSERT INTO branch VALUES(1,"Chennai","16 ABC Road");
INSERTINTObranchVALUES(2,"Coimbatore","12015thBlock");
INSERT INTO branch VALUES(3,"Mumbai","25 XYZ Road");
INSERTINTObranchVALUES(4,"Hydrabad","3210thStreet");

INSERT INTO employee VALUES(1,'Ram','ADMIN',1000000,2);
INSERTINTOemployeeVALUES(2,'Harini','MANAGER',2500000,2);
INSERT INTO employee VALUES(3,'George','SALES',2000000,1);
INSERT INTO employee VALUES(4,'Ramya','SALES',1300000,2);
INSERT INTO employee VALUES(5,'Meena','HR',2000000,3);INSERT
INTO employee VALUES(6,'Ashok','MANAGER',3000000,1); INSERT
INTO employee
VALUES(7,'Abdul','HR',2000000,1);INSERTINTOemployeeVALUES(8,
'Ramya','ENGINEER',1000000,2); INSERT INTO employee
VALUES(9,'Raghu','CEO',8000000,3);
INSERTINTOemployeeVALUES(10,'Arvind','MANAGER',2800000,3);
INSERTINTOemployeeVALUES(11,'Akshay','ENGINEER',1000000,1);
INSERT INTO employee VALUES(12,'John','ADMIN',2200000,1);
INSERTINTOemployeeVALUES(13,'Abinaya','ENGINEER',2100000,2);
INSERTINTOemployeeVALUES(14,'Vidya','ADMIN',2200000,NULL);
INSERTINTOemployeeVALUES(15,'Ranjani','ENGINEER',2100000,NULL);

SELECT*FROMemployee;
SELECT * FROM branch;

--innerjoin:onlymatchingrows
SELECT employee.emp_id,employee.ename,employee.job_desc,branch.br_name
FROM employee
INNERJOINbranch
ON employee.branch_id=branch.branch_id
ORDER BY emp_id;

--belowquerygivessameresultwithoutusingjoinkeyword
SELECT employee.emp_id,employee.ename,employee.job_desc,branch.br_name
FROM employee,branch
WHERE employee.branch_id=branch.branch_id
ORDER BY emp_id;

--usingtablenamealias
SELECT e.emp_id,e.ename,e.job_desc,b.br_name
```

```
INNERJOINbranchASb
ON e.branch_id=b.branch_id
ORDER BY e.emp_id;

--Rightjoinismatchedrows+allotherrowsinrighttable
SELECT employee.emp_id,employee.ename,employee.job_desc,branch.br_name
FROM employee
RIGHTJOINbranch
ON employee.branch_id=branch.branch_id
ORDER BY emp_id;

--Leftjoinismatchedrowswithallotherrowsinlefttable
SELECT employee.emp_id,employee.ename,employee.job_desc,branch.br_name
FROM employee
LEFTJOINbranch
ON employee.branch_id=branch.branch_id
ORDER BY emp_id;

--Crossjoinjoinseachrowoffirsttablewitheveryotherrowofsecondtable
SELECTemployee.emp_id,employee.ename,employee.job_desc,branch.br_name
FROM employee
CROSSJOINbranch;

--displaystheemployeecountineachbranch SELECT
b.br_name,COUNT(e.emp_id)
FROM branch as b
JOINemployeease
ONb.branch_id=e.branch_id
GROUP BY e.branch_id;
```

Union

unioncombinestwotablehavingequalnumberofcolumnsandmatchingdatatypes

```
--createclienttablesimilartobranchtable
CREATE TABLE clients (
client_idINTPRIMARYKEYAUTO_INCREMENT,
location VARCHAR(30) NOT NULL,
addrVARCHAR(200));

INSERT INTO clients VALUES(1,"NewYork","25 10th Block");
INSERTINTOclientsVALUES(2,"Coimbatore","12015thBlock");
INSERT INTO clients VALUES(3,"London","21 ABC Road");

--combinesthetwotablesremovingduplicates
SELECT * FROM branch
UNION
SELECT*FROMclients;

--combinesthetwotableswithoutremovingduplicates SELECT *
FROM branch
UNIONALL
SELECT*FROMclients;
```

Subqueries,Exists,Any,All

Subqueriescombinemorethan2queries.

```
--DisplaysemployeelistinChennaiBranch SELECT
* FROM employee
WHEREbranch_id=(SELECTbranch_id
FROM branch
WHEREbr_name="Chennai");

--Displaystheemployeeswithminsalary
SELECT * FROM employee
WHEREsalary=(SELECTMIN(salary)
FROM employee);

  -displaysthebranchescontainingatleastoneadmin
```

```
SELECTbranch_id,br_name
FROM branch
WHEREEXISTS
(SELECT*FROMemployee
WHEREjob_desc="ADMIN"ANDbranch.branch_id=employee.branch_id);

--displaysthebranchinfoinwhichanyemployeegetsmorethan25L SELECT
branch_id,br_name
FROMbranch
WHEREbranch_id=ANY
(SELECTbranch_idFROMemployee
WHERE salary>2500000);

--displaysemployeesnotworkinginchennaiorcoimbatore

SELECT*FROMemployee
WHEREbranch_id<>ALL(SELECTbranch_idFROMbranch WHERE
br_name IN ("Chennai","Coimbatore"));
```

Views

```
CREATEVIEWemp_br
AS
SELECT employee.emp_id,employee.ename,employee.job_desc,branch.br_name
FROM employee
INNERJOINbranch
ON employee.branch_id=branch.branch_id
ORDER BY emp_id;

SELECT*FROMemp_br;--selectingallrowsfromview DROP VIEW

emp_br; -- delete view

CREATEORREPLACEVIEWemp_br--modifyview AS
SELECT employee.emp_id,employee.ename,employee.job_desc,branch.br_name
FROM employee
INNERJOINbranch
ONemployee.branch_id=branch.branch_id;
```