SQL

Introduction

SQL Structure

Introduction

What is a Database?

Design

Database Architecture

Oracle Database Structure

Setup Connection

IDEs

SQL

Wisdom
Intelligence
Knowledge
Learning
Information
Relations
Data

capture
Data
relate
Wisdom
Information
Knowledge
apply
learn

The Big Picture of Data Management

Generate data
Extract
Transform
Load
Create
Database
Query
Manipulate
Show
value
BMS
IoT
CostIt
Costs
DB
Models
XLSX
Others
ETL
DW
Dashboards

What is a database?

A structured set of data according to a certain data model and stored in a computer media, accessible to people and applications.
(diccionario.raing.es)

Autor: tedeytan Fuente: http://flickr.com/photos/22526649@N03/5466788868

Database Design

Information
We need to store it
DB

Information
Relational Model
SQL
Relational design
RDBMS
DB
Normalisation
1 NF
2 NF
3 NF

Conceptual level

Logical level

Physical level

Conceptual Model

We want to store information about departments: name and location.

For employees: name, salary, job, and the boss.

An employee is in a department.

There can be several employees in a department.

An employee only has one boss.

A boss can supervise several employees.

Entity-Relationship Model

We want to store information about departments: name and location.

For employees: name, salary, job, and the boss.

An employee is in a department.

There can be several employees in a department.

An employee only has one boss.

A boss can supervise several employees.

Entity-Relationship Model

We want to store information about departments: name and location.

For employees: name, salary, job, and the boss.

An employee is in a department.

There can be several employees in a department.

An employee only has one boss.

A boss can supervise several employees.

Entity-Relationship Model

We want to store information about departments: name and location.

For employees: name, salary, job, and the boss.

An employee is in a department.

There can be several employees in a department.

An employee only has one boss.

A boss can supervise several employees.

Entity-Relationship Model

We want to store information about departments: name and location.

For employees: name, salary, job, and the boss.

An employee is in a department.

There can be several employees in a department.

An employee only has one boss.

A boss can supervise several employees.

Relational Model

Deptno	Dname	Loc
10		
20		
30		
40		

Empno	Sal	...
7878		
7879		
7979		
7576		
7675		

Relational Model

Deptno	Dname	Loc
10		
20		
30		
40		

Empno	Sal	...	Deptno
7878			10
7879			20
7979			10
7576			30
7675			

Relational Model

Deptno	Dname	Loc
10		
20		
30		
40		

Empno	Sal	...	Deptno	Mgr
7878			10	
7879			20	7878
7979			10	7878
7576			30	7979
7675				7675

Architecture

Autor: Oracle Databases Fuente: http://www.flickr.com/photos/12949199@N00/1507478972/ Oracle Databases

Server

Oracle
MySQL
DB2
SQL Server

Client-Server

Web

Database Server
Application
DB
RDBMS
Applications Server
Accounting
Sales
HR
Billing
App
App
Internet Explorer
Mozilla Firefox
Chrome

RDBMS Objectives

- Physical/logical data independence
- Utilisation of a standard language for accessing information
- Data persistence
- Centralised Data Access
- Concurrency control
- Data Access security
- Efficiency in CPU, memory, and I/O processing
- Data integrity

Oracle Structure

Tablespaces

Schemes

Connection

Autor: Justin Fuente: http://flickr.com/photos/17964151@N00/3677688990

Network configuration

netca

Services and Files

lnsrctl

OracleDBConsolejp		Manual
OracleJobSchedulerJP		Deshabilitado
OracleJobSchedulerORCL		Deshabilitado
OracleMTSRecoveryService		Manual
OracleOraDb10g_home1iSQL*Plus	iSQL*Plus ...	Manual
OracleOraDb10g_home1TNSListener		Manual
OracleOraDb11g_home1ClrAgent		Manual
OracleOraDb11g_home1TNSListener		Manual
OracleServiceJP		Manual
OracleServiceORCL		Manual

$ORACLEHOME/NETWORK/ADMIN

LISTENER.ORA
TNSNAMES.ORA

IDEs

SQLPLUS

sqlplus username/password@serviceName

SQL Developer

SQL

Autor: amortize Fuente: https://www.flickr.com/photos/amortize/527435776

SQL

Structured Query Language

SQL is a query language, not a programming language.

Data Types

NUMBER

NUMBER	any real number
NUMBER (n)	n integer digits
NUMBER (n,m)	n digits, m are decimals

VARCHAR2

VARCHAR2 (n)	n length

DATE

SQL Structure

SELECT

DML: INSERT, UPDATE, DELETE, MERGE

Transactional Statements:
ROLLBACK, COMMIT

DDL: CREATE, ALTER, DROP, RENAME , TRUNCATE.

DCL: GRANT, REVOKE.

SELECT

SELECT

```
SELECT   [ALL|DISTINCT]
{ *  |  {column | expression} [[AS] alias], ... }
FROM
   {[schema.]{table|view} | (sub-
query)}[alias][, …]
[WHERE          conditions_list]
[GROUP BY       columns_list]
[HAVING         group_conditions]
[ORDER BY       column [ASC|DESC]]
;
```

SELECT

```
SELECT *
FROM   emp;
```

SELECT

```
SELECT ename, sal, sal+comm SalTotal
FROM   emp;
```

SELECT

```
SELECT ename, sal, sal+comm SalTotal
FROM   emp
WHERE  deptno=20;
```

SELECT

```
SELECT ename, sal, sal+comm SalTotal
FROM   emp
WHERE  deptno=20
ORDER BY  1 ASC, 3 DESC;
```

CASE

```
SELECT ename,
       CASE deptno
          WHEN 10 THEN
             'Ten'
          WHEN 20 THEN
             'Twenty'
          ELSE
             'Other'
       END depart
FROM   emp;
```

CASE

```
SELECT ename,
   CASE
      WHEN deptno=10 THEN
         'Ten'
      WHEN deptno=20 THEN
         'Twenty'
      ELSE
         'Other'
   END depart
FROM    emp;
```

Conditions

```
WHERE cond1
      AND ((cond2 OR cond3)
            AND cond4)
```

=	IN
>	ALL
<	BETWEEN
>=	LIKE
<=	NOT
<>	NULL
!=	
~=	

Predefined Functions

ROUND	LENGTH	SYSDATE	TO_NUMBER
TRUNC	UPPER	USER	TO_DATE
NVL	LOWER		TO_CHAR
COUNT	INITCAP		
MAX	RPAD		
MIN	LPAD		
SUM	SUBSTR		
AVG			

GROUP BY

```
SELECT      SUM(sal), deptno
FROM        emp
GROUP BY    deptno;
```

SUM
MAX
MIN
COUNT
AVG

HAVING

```
SELECT      SUM(sal), deptno
FROM        emp
GROUP BY    deptno
HAVING      SUM(sal) > 2500;
```

SUM
MAX
MIN
COUNT
AVG

JOINS

```
SELECT
      EMP.deptno,empno,dname
FROM   EMP, DEPT
WHERE
      EMP.deptno=DEPT.deptno;
```

FK PK

Empno	Sal	...	Deptno
7878			10
7979			20
7676			10
7575			30
7474			

Deptno	Dname	Loc
10	SALES	
20	ADMIN	
30	ACCOUNT	
40	EXECU	

JOINS

Empno	Sal	...	Deptno
7878			10
7979			20
7676			10
7575			30
7474			

Deptno	Dname	Loc
10	SALES	
20	ADMIN	
30	ACCOUNT	
40	EXECU	

EMP.deptno	empno	dname
10	7878	SALES
20	7979	ADMIN
10	7676	SALES
30	7575	ACCOUNT

JOINS

```
SELECT
      EMP.deptno,ename,dname
FROM    EMP, DEPT
WHERE EMP.deptno=DEPT.deptno
      AND sal>3000;
```

JOINS

```
SELECT E.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE E.deptno = D.deptno
      AND sal > 3000;
```

JOINS

```
SELECT E.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE  E.deptno = D.deptno(+);
```

Empno	Sal	...	Deptno
7878			10
7979			20
7676			10
7575			30
7474			

Deptno	Dname	Loc
10	SALES	
20	ADMIN	
30	ACCOUNT	
40	EXECU	

JOINS

Empno	Sal	...	Deptno
7878			10
7979			20
7676			10
7575			30
7474			

Deptno	Dname	Loc
10	SALES	
20	ADMIN	
30	ACCOUNT	
40	EXECU	

E.deptno	empno	dname
10	7878	SALES
20	7979	ADMIN
10	7676	SALES
30	7575	ACCOUNT
	7474	

JOINS

```
SELECT E.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE E.deptno = D.deptno(+);
```

E.deptno	empno	dname
10	7878	SALES
20	7979	ADMIN
10	7676	SALES
30	7575	ACCOUNT
	7474	

JOINS

```
SELECT E.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE E.deptno(+) = D.deptno;
```

E.deptno	empno	dname
10	7878	SALES
20	7979	ADMIN
10	7676	SALES
30	7575	ACCOUNT
		EXECU

JOINS

Empno	Sal	...	Deptno
7878			10
7979			20
7676			10
7575			30
7474			

Deptno	Dname	Loc
10	SALES	
20	ADMIN	
30	ACCOUNT	
40	EXECU	

E.deptno	empno	dname
10	7878	SALES
20	7979	ADMIN
10	7676	SALES
30	7575	ACCOUNT
		EXECU

JOINS

```
SELECT E.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE E.deptno(+) = D.deptno;
```

E.deptno	empno	dname
10	7878	SALES
20	7979	ADMIN
10	7676	SALES
30	7575	ACCOUNT
		EXECU

JOINS

```
SELECT D.deptno, ename, dname
FROM   EMP E, DEPT D;
```

```
SELECT D.deptno, ename, dname
FROM EMP CROSS JOIN DEPT D;
```

JOINS

```
SELECT D.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE  E.deptno=D.deptno;
```

```
SELECT deptno, ename, dname
FROM   EMP JOIN DEPT
       USING(deptno);
```

JOINS

```
SELECT D.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE E.deptno=D.deptno;
```

```
SELECT D.deptno, ename, dname
FROM   EMP E JOIN DEPT D
  ON(E.deptno=D.deptno);
```

JOINS

```
SELECT D.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE  E.deptno(+)=D.deptno;
```

```
SELECT deptno, ename, dname
FROM   EMP RIGHT JOIN DEPT
       USING(deptno);
```

JOINS

```
SELECT D.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE  E.deptno=D.deptno(+);
```

```
SELECT deptno, ename, dname
FROM   EMP LEFHT JOIN DEPT
       USING(deptno);
```

JOINS

```
SELECT D.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE E.deptno(+)=D.deptno(+);
```

```
SELECT deptno, ename, dname
FROM   EMP FULL JOIN DEPT
       USING(deptno);
```

JOINS

```
SELECT D.deptno, ename, dname
FROM   EMP E, DEPT D
WHERE  E.deptno(+)=D.deptno(+);
```

```
SELECT deptno, ename, dname
FROM   EMP FULL JOIN DEPT
       USING(deptno);
```

Sub-queries

```
SELECT   ename, sal
FROM     emp
WHERE    deptno IN
                (SELECT deptno
                FROM     dept
                WHERE loc='DALLAS');
```

Sub-queries

```
SELECT        deptno
FROM          emp
GROUP BY      empno
HAVING  COUNT(empno) =
        (SELECT  MIN(COUNT(empno))
        FROM     emp);
```

Sub-queries

```
SELECT   ename,sal,loc,average
FROM
      (SELECT ename,sal,loc
      FROM emp
         JOIN dept USING (deptno)) t1
   JOIN
      (SELECT AVG(sal) average,loc
      FROM emp
         JOIN dept USING (deptno)
      GROUP BY loc) t2
   USING (loc);
```

Sub-queries

```
SELECT   deptno,
         (SELECT  SUM(sal)
         FROM     emp
         WHERE    deptno=D.deptno) SumSal
FROM     dept D;
```

DML

Autor: Raidarmax Fuente: https://upload.wikimedia.org/wikipedia/commons/4/4a/Numark_Mixtrack_Pro_DJ_Controller.jpg

INSERT

```
INSERT INTO dept
VALUES(50,'TC','MADRID')
;
```

INSERT

```
INSERT INTO dept(
    deptno,
    dname)
VALUES(
    60,
    (SELECT department_name
    FROM departments
    WHERE department_id=260)
        );
```

INSERT

```
INSERT INTO dept
   SELECT      department_id,
               department_name,
               city
   FROM  departments
      JOIN locations USING (location_id)
   WHERE department_id IN (80,90)
```

DELETE

```
DELETE dept
WHERE deptno=60;
```

UPDATE

```
UPDATE emp
SET    sal=3000
WHERE  deptno=20;
```

UPDATE

```
UPDATE  emp E
SET  sal=(SELECT   AVG(sal)
          FROM     emp
          WHERE    deptno=E.deptno);
```

MERGE

```
MERGE INTO empleados e1
USING emp e2
ON(e1.empno=e2.empno)
WHEN MATCHED THEN
   UPDATE SET e1.sal=e2.sal, e1.job=e2.job
      WHERE e1.deptno IN (20,30)
   DELETE
      WHERE e1.empno=7521
WHEN NOT MATCHED THEN
   INSERT (e1.empno, e1.ename, e1.deptno)
   VALUES(e2.empno, e2.ename, e2.deptno)
   WHERE e2.hiredate IS NOT NULL;
```

Transactional Statements

Transactional Statements

COMMIT

ROLLBACK

SAVE POINT

DDL

Constraint Types

PRIMARY KEY
FOREIGN KEY
UNIQUE
CHECK

PRIMARY KEY

Name
Maria
MARIA
María
María

PRIMARY KEY

Name	Surname
Maria	Lopez
MARIA	López
María	Lopez
Mary	
María	López

FOREIGN KEY

Empno	Sal	...	Deptno
7878			10
7879			20
7979			10
7576			30
7675			
~~7777~~			~~77~~

Deptno	Dname	Loc
10		
20		
30		
40		

UNIQUE

Name
Maria
MARIA
María
~~María~~

CREATE TABLE

```
CREATE TABLE emp_20
AS
    SELECT
        empno,
        ename,
        deptno
    FROM emp
    WHERE deptno=20;
```

CREATE TABLE

```
CREATE TABLE phones(
phone       VARCHAR2 (14),
label       VARCHAR2 (128),
user        NUMBER(4),
created     DATE DEFAULT SYSDATE NOT NULL
CONSTRAINT pk_phones_num
      PRIMARY KEY (phone),
CONSTRAINT uq_phones_label
      UNIQUE(label));
```

ALTER TABLE

```
ALTER TABLE phones ADD (
CONSTRAINT fk_phones_user
        FOREIGN KEY(user_id),
    REFERENCES emp (empno)
);
```

ALTER TABLE

```
ALTER TABLE phones DROP
CONSTRAINT uq_phones_label
);
```

ALTER TABLE

```
ALTER TABLE phones ADD (
updated_at          DATE,
updated_by      NUMBER(4)
);
```

ALTER TABLE

```
ALTER TABLE phones DROP (
updated_at,
updated_by);
```

Constraints

- **Disabled**

```
ALTER TABLE        table_name
DISABLE CONSTRAINT constraint_name;
```

- **Enforced**

```
ALTER TABLE        table_name
ENFORCE CONSTRAINT constraint_name;
```

- **Actived**

```
ALTER TABLE        table_name
ENABLE VALIDATE CONSTRAINT constraint_name;
```

DCL

Autor: Mr Bullitt Fuente: https://upload.wikimedia.org/wikipedia/commons/a/a7/United_Nations_Security_Council.JPG

DCL

GRANT

REVOKE

GRANT

```
GRANT select, update
ON emp
TO student3, student4;

GRANT
        connect,
        resources,
        debug connect session,
        debug any procedure
TO
        student3, student4;
```

REVOKE

```
REVOKE select
ON emp
FROM student3, student4;

REVOKE
    dba
FROM
    student3;
```

ROLE

```
CREATE ROLE my_role;


GRANT select ON emp TO my_role;
GRANT select ON dept TO my_role;


GRANT
    my_role
TO
    student3;
```

License

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License. To view a copy of this license, visit: http://creativecommons.org/licenses/by-nc/4.0/.

You are free to:

Share — copy and redistribute the material in any medium or format
Adapt — remix, transform, and build upon the material

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give **appropriate credit**, provide a link to the license, and **indicate if changes were made**. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
NonCommercial — You may not use the material for **commercial purposes**.

Updated: 2024-06-14