Advantages of DBMS

1. Reduced application dev time
   a. Uniform data administration
2. Data independence
   a. Users don’t need to know how data stored on disk
   b. DBA can change how data stored without affecting application
3. Efficient access
   a. Users tell DBMS what they want, but now how they get the data
   b. DBMS finds most efficient way to retrieve data
4. Data integrity
   a. DBMS checks if data entered satisfy constraints set by DBA
   b. DBMS performs crash recovery to make sure data is correct
5. Ease of data sharing and security
   a. DBA tells DBMS which user have privileges

When not to use DBMS

- When read speed need to be very fast
- Single user, single machine

Disadvantage

- Data manipulation not so easy if using query languages

DDL (Data Definition language)

-enables creation, deletion, modification, definitions of tables

DML (Data manipulation language)

-allows users to query , insert, delete , or update rows

Creating databases example

```
Create database if not exists threetables;
Use threetables;
Drop table if exists `works`;
Drop table if exists `dept`;
Drop table if exists `emp`;

Create table `emp`(
`eid` int(11),
`ename` varchar(50) Not null,
`salary` decimal(10,0) default null,
Primary key (`eid`)
);

CREATE TABLE `dept` (
`did` int(11),
`dname` varchar(40) NOT NULL,
`budget` decimal(10,0) DEFAULT 100000,
`managerid` int DEFAULT NULL,
PRIMARY KEY (`did`),
CONSTRAINT `managerid` FOREIGN KEY (`managerid`) REFERENCES `emp` (`eid`) ON DELETE SET NULL
);
```

```
CREATE TABLE `works` (
      `eid` int(11),
      did` int(11),
      `pct_time` int(11) DEFAULT 10,
      PRIMARY KEY (`eid`,`did`),
      CONSTRAINT `eid` FOREIGN KEY (`eid`) REFERENCES `emp` (`eid`),
      CONSTRAINT `did` FOREIGN KEY (`did`) REFERENCES `dept` (`did`)
);
```

Inserting values into database

```
INSERT INTO `dept` VALUES
(1,'InformationTechnology',1500000,NULL),(2,'Production',2000000,105),(3,'Administration',2000000,105);
```

Different kinds of joins

To get the cross product of two table (mashing everything together)

- Select * from dept, work

To get rows with matching id’s

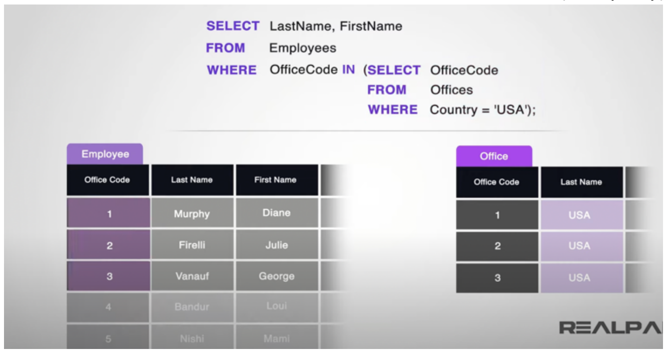
- Select * from emp e inner join dept d on e.eid = d.did

Left join (right join works similarly)

- Takes all rows from left table, only matching on the right (with where clause)

In operator (can use not in too)

Format: Select attributes from table1 where attribute in (Subquery)



```
SELECT  LastName, FirstName
FROM    Employees
WHERE   OfficeCode IN (SELECT  OfficeCode
                       FROM    Offices
                       WHERE   Country = 'USA');
```

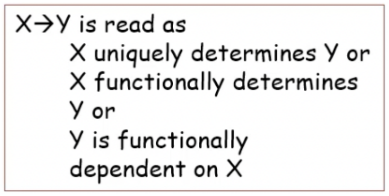
Employee

Office Code	Last Name	First Name
1	Murphy	Diane
2	Firelli	Julie
3	Vanauf	George
4	Bandur	Loui
5	Nishi	Mami

Office

Office Code	Last Name
1	USA
2	USA
3	USA

Functional dependency

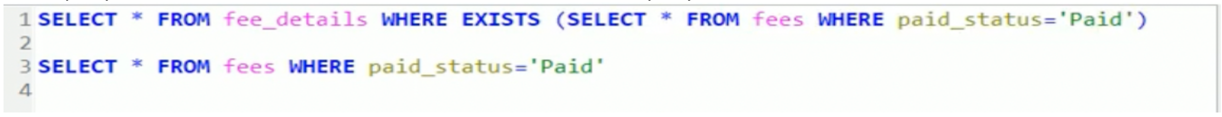


X→Y is read as
X uniquely determines Y or
X functionally determines Y or
Y is functionally dependent on X

Exists operator (can use not exists too)

Format: Select attribute from table 1 where exists (subquery)

If subquery returns 1 or more result, Boolean value is true, outer query will run



```
SELECT * FROM fee_details WHERE EXISTS (SELECT * FROM fees WHERE paid_status='Paid')

SELECT * FROM fees WHERE paid_status='Paid'

```

Groupby operator and having (often used together)

Select eid, count(*) from works group by eid having count(*) >2;

Redundancy issues

- Insertion anomaly (cant insert rows with null values)
- Deletion anomaly (we lose more data than intended when we only delete some attribute of a row)
- Update anomaly (if we update value of an attribute, we must do it for the copies too)

Helpful commands

Selecting only a certain number of rows

Select * from table LIMIT 3;

Selecting rows based on ascending or descending order

Select * from table order by <attribute> ASC;   Select * from table order by <attribute> DSC;

Select min or max of an attribute from a table

Select MIN(Price) As smallestPrice from Products; Select MAX(Price) As largestPrice from Products;

Treating a subquery as a table

Select eid From (Select eid FROM emp_work_dept where dname = "Production") as Prod
Inner Join (Select eid as m_id FROM emp_work_dept where dname = "Maintenance") as Main
Where Prod.eid = Main.m_id;

Select distinct rows from a table

Select distinct (ename) from emp;

Select attributes where it is not null

Select distinct did, dname FROM emp_work_dept Where managerid IS NOT NUll;

```
select sid
from catalog
where pid in (select pid
 from parts where color='green')
group by sid
having count(pid) = (
select count(*)
from parts
where color='green')
order by sid;

-- using the exists operator
select c.sid
from catalog c
where exists (select *
 from parts where color='green' and c.pid=pid)
group by sid
having count(pid) = (
select count(*)
from parts
where color='green')
order by sid;
```