Bases de Datos

Hector Florez
haflorezf@udistrital.edu.co

Contenido

- Introducción
  - Definiciones
  - Modelos
  - Lenguajes
- Modelo Entidad-Relación
- Modelo Relacional
- Diseño de Bases de Datos Relacionales
  - Normalización
- Lenguajes Formales de Consulta
  - Algebra Relacional

Contenido

- Introducción a SQL
  - DDL
  - DML
- SQL Intermedio
  - Join
  - Vistas
  - Subconsultas anidadas
- SQL Avanzado
  - Funciones
  - Disparadores
  - Consultas Recursivas
  - Agregación Avanzada

Evaluación

- Laboratorios
  - 20%
- Examen 1 (Traer hoja examen)
  - 15%
  - Marzo 20
- Proyecto 1
  - 20%
  - Abril 10
- Examen 2
  - 15%
  - Mayo 22
- Proyecto 2
  - 30%
  - Mayo 29

Bibliografía

- Abraham Silberschatz, Henry F. Korth, S. Sudarshan. DATABASE SYSTEM CONCEPTS. Sixth Edition. 2011

Introducción

Introducción

- **Base de datos:**
  - Colección de datos almacenados en archivos que contiene información relevante de un negocio.
  - Permite manejar grandes cantidades de información.
- **Sistema manejador de bases de datos (DBMS):**
  - Colección de programas que permite acceder a los datos almacenados en una base de datos.
- **Aplicaciones transaccionales - Aplicaciones de bases de datos - Sistemas de información.**
  - Aplicaciones en contextos como: comercio, banca, academico, etc.

Introducción

- **Propósitos de las base de datos:**
  - Mantener la información organizada
  - Asegurar las siguientes características:
    - ACID
      - Atomicidad, Consistencia, Aislamiento, Disponibilidad
    - Facilidad de acceso
    - Integridad
    - Concurrencia
    - Seguridad

Introducción

- **Niveles de abstracción de datos**
  - Nivel Físico
    - Describe cómo los datos están almacenados
  - Nivel Lógico
    - Describe qué datos están almacenados y qué relaciones existen entre los datos
  - Nivel de visualización
    - Describe las formas de presentar los datos al usuario

Introducción

- **Modelos**
  - Modelo Entidad - Relación
    - Usa objetos denominados entidades y relaciones
    - Una entidad es la representación de un concepto de la realidad
  - Modelo Relacional
    - Usa tablas para representar entidades y relaciones
    - Cada tabla tiene múltiples columnas con único nombre
    - Es el modelo más utilizado

Introducción

- **Lenguajes**
  - Data Manipulation Language (DML)
    - Permite acceder y manipular datos en una forma organizada
    - Los tipos de acceso son:
      - Creación.
      - Consulta.
      - Actualización
      - Eliminación
  - Data Definition Language (DDL)
    - Especifica la estructura y métodos de acceso de la base de datos

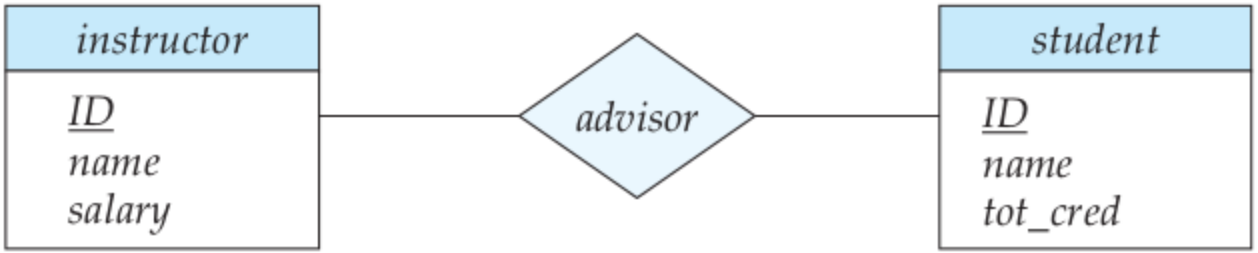
Modelo Entidad-Relación

Modelo Entidad-Relación

- Es un modelo conceptual que se basa de Entidades y Relaciones
  - Una entidad es una representación de un objeto de la realidad
    - Contiene atributos que describe las propiedades del objeto representado
  - Una relación es una asociación entre entidades
- Un modelo E-R además tiene:
  - Cardinalidad
    - 1 a 1
    - 1 a muchos
    - Muchos a 1
    - Muchos a muchos

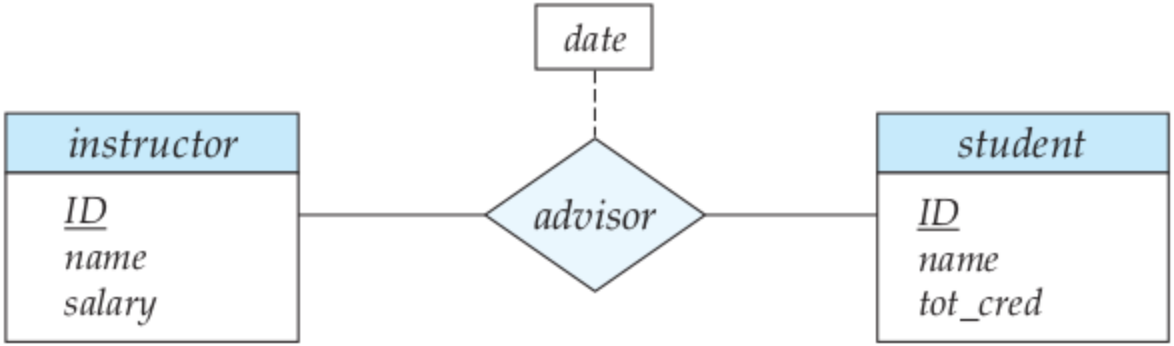
Modelo Entidad-Relación

- Diagrama Entidad-Relación



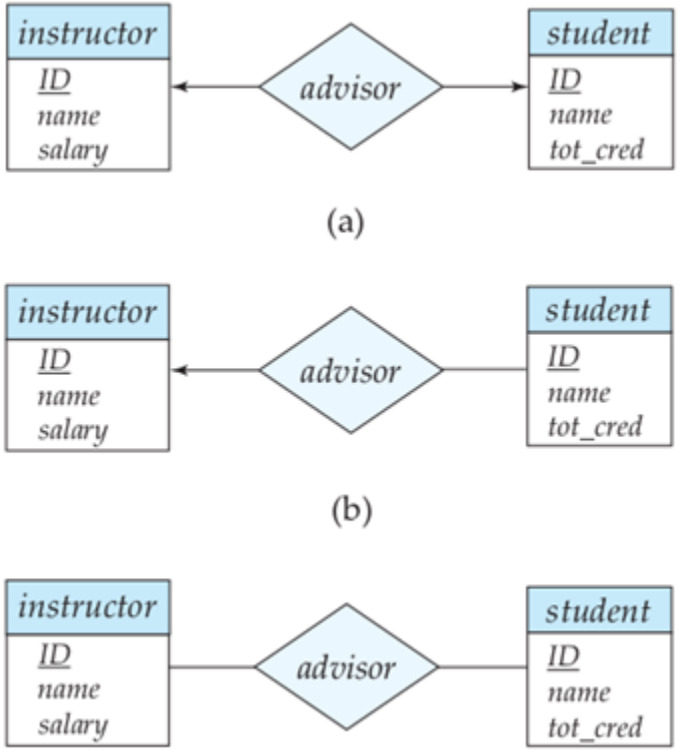
Modelo Entidad-Relación

- Diagrama Entidad-Relación



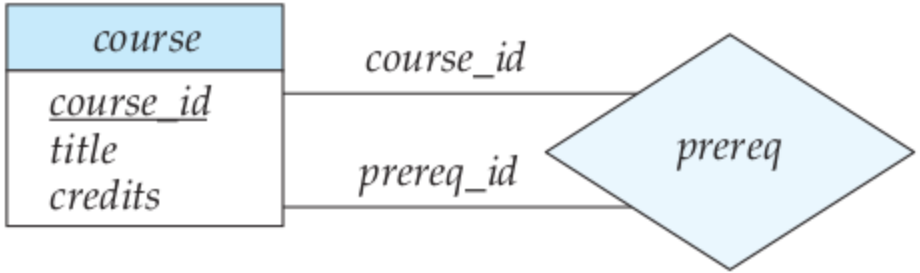
Modelo Entidad-Relación

- Diagrama Entidad-Relación



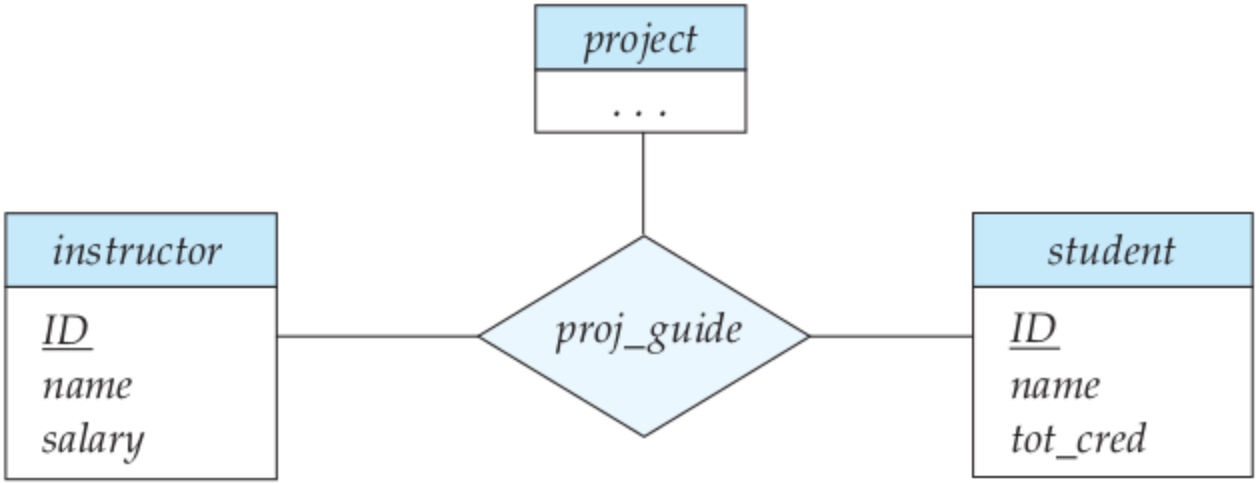
Modelo Entidad-Relación

- Diagrama Entidad-Relación



Modelo Entidad-Relación

- Diagrama Entidad-Relación

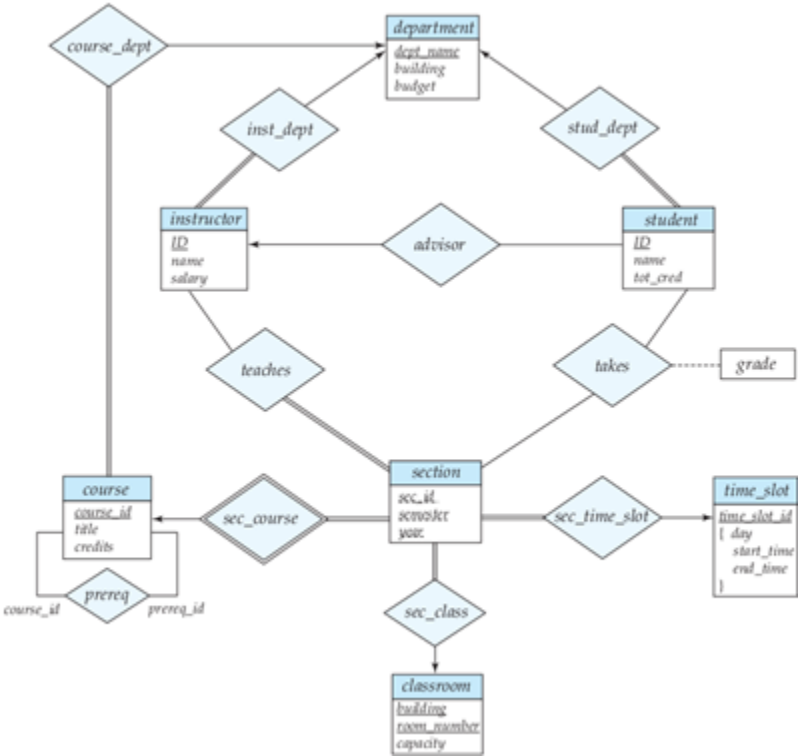


Modelo Entidad-Relación

- Diagrama Entidad-Relación

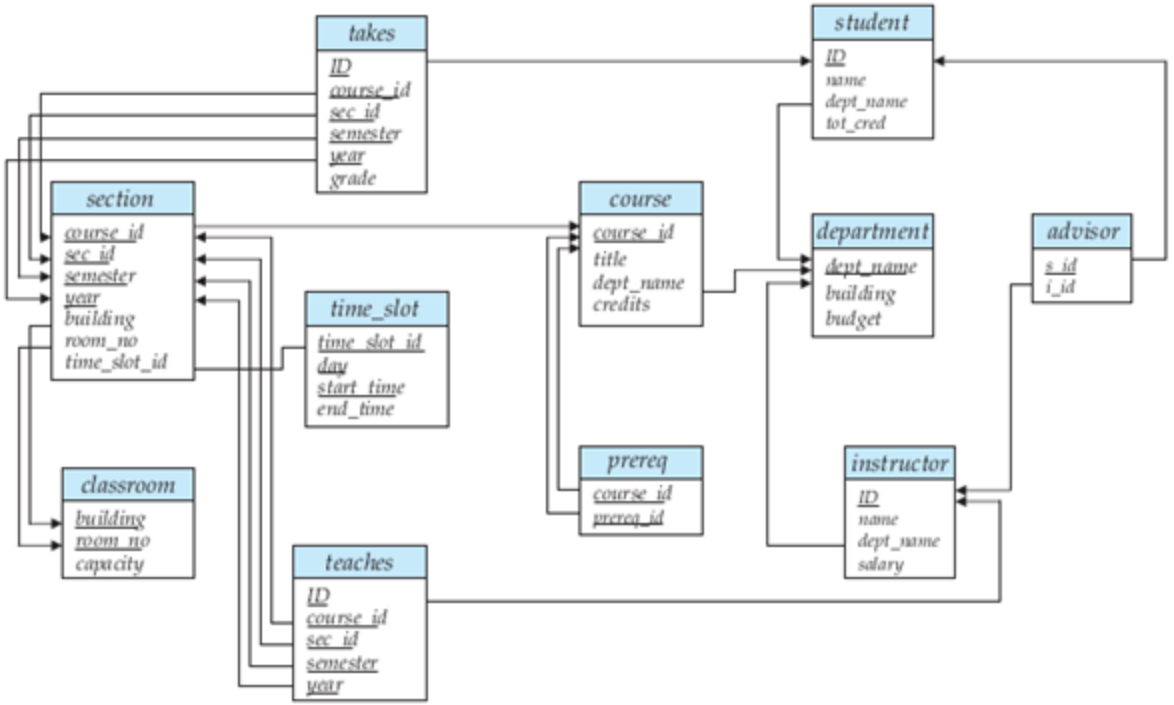
Modelo Entidad-Relación

- Diagrama Entidad-Relación



Modelo Relacional

Modelo Relacional



Diseño de Bases de Datos Relacionales

Diseño de Bases de Datos Relacionales

- Normalización
  - Proceso para estructurar una base de datos mediante formas normales
  - Organiza tablas y atributos para asegurar dependencias adecuadas respecto a restricciones de integridad
  - Reduce redundancia
  - Mejora integridad de los datos

Diseño de Bases de Datos Relacionales

- Primera forma normal
  - Una BD se encuentra en primera forma normal si:
    - Contiene sólo valores atómicos

id	nombre	apellido	telefono
10	Clark	Kent	123, 456
20	Bruce	Wayne	654, 321

id	nombre	apellido	telefono
10	Clark	Kent	123
10	Clark	Kent	456
20	Bruce	Wayne	654
20	Bruce	Wayne	321

Diseño de Bases de Datos Relacionales

- Segunda forma normal
  - Una BD se encuentra en segunda forma normal si:
    - Está en primera forma normal
    - Cada atributo depende completamente de la llave primaria.
    - No tiene dependencias funcionales (atributo que depende de más de un candidato a llave primaria)

Diseño de Bases de Datos Relacionales

id	nombre	apellido	ciudad
10	Clark	Kent	Bogota
20	Bruce	Wayne	Bogota D.C.
30	Petter	Parker	Medellin
40	Tony	Stark	Quito

id	nombre	apellido	idCiudad
10	Clark	Kent	100
20	Bruce	Wayne	100
30	Petter	Parker	200
40	Tony	Stark	300

id	ciudad
100	Bogota
200	Medellin
300	Quito

Diseño de Bases de Datos Relacionales

- Tercera forma normal
  - Una BD se encuentra en tercera forma normal si:
    - Está en segunda forma normal
    - No tiene dependencias funcionales transitivas

Diseño de Bases de Datos Relacionales

id	nombre	apellido	ciudad	pais
10	Clark	Kent	Bogota	Colombia
20	Bruce	Wayne	Bogota	Colombia
30	Petter	Parker	Medellin	Colombia
40	Tony	Stark	Quito	Ecuador

id	nombre	apellido	idCiudad
10	Clark	Kent	100
20	Bruce	Wayne	100
30	Petter	Parker	200
40	Tony	Stark	200

id	ciudad	idPais
100	Bogota	1000
200	Medellin	1000
300	Quito	2000

id	pais
1000	Colombia
2000	Ecuador

Diseño de Bases de Datos Relacionales

- Cuarta forma normal
  - Una BD se encuentra en cuarta forma normal si:
    - Está en tercera forma normal
    - No tiene dependencias funcionales multivaluadas

Diseño de Bases de Datos Relacionales

Restaurante	Variedad de Pizza	Área de envío
Dominos Calle 53	Pollo y Champiñones	Chapinero
Dominos Calle 53	Pollo y Champiñones	Teusaquillo
Dominos Calle 53	Hawaiana	Chapinero
Dominos Calle 53	Hawaiana	Teusaquillo
Dominos Calle 26	Pollo y Champiñones	Candelaria
Dominos Calle 26	Pollo y Champiñones	Chapinero
Dominos Calle 26	Pollo y Champiñones	Teusaquillo
Dominos Calle 26	Hawaiana	Candelaria
Dominos Calle 26	Hawaiana	Chapinero
Dominos Calle 26	Hawaiana	Teusaquillo

Diseño de Bases de Datos Relacionales

Restaurante	Variedad de Pizza
Dominos Calle 53	Pollo y Champiñones
Dominos Calle 53	Hawaiana
Dominos Calle 26	Pollo y Champiñones
Dominos Calle 26	Hawaiana

Restaurante	Área de envío
Dominos Calle 53	Chapinero
Dominos Calle 53	Teusaquillo
Dominos Calle 26	Candelaria
Dominos Calle 26	Chapinero
Dominos Calle 26	Teusaquillo

Lenguajes Formales de Consulta

Lenguajes Formales de Consulta

- Algebra Relacional
  - Es un lenguaje de consulta
  - Permite realizar operaciones que toma entidades y relaciones y produce una nueva relación
- Caso de Estudio online
- Operaciones
  - Select

$$\sigma_{dept\_name = \text{“Physics”}} (instructor)$$

$$\sigma_{salary > 90000} (instructor)$$

$$\sigma_{dept\_name = \text{“Physics”} \wedge salary > 90000} (instructor)$$

$$\sigma_{dept\_name = building} (department)$$

Lenguajes Formales de Consulta

- Operaciones
  - Project

$$\Pi_{ID,\ name,\ salary}(instructor)$$

  - Composition

$$\Pi_{name}\ (\sigma_{dept\_name = \text{“Physics”}}\ (instructor))$$

  - Union

$$\Pi_{course\_id}\ (\sigma_{semester = \text{“Fall”} \wedge year = 2009}\ (section)) \cup \Pi_{course\_id}\ (\sigma_{semester = \text{“Spring”} \wedge year = 2010}\ (section))$$

Lenguajes Formales de Consulta

- Operaciones
  - Set-Difference

$$\Pi_{course\_id}\ (\sigma_{semester = \text{“Fall”} \wedge year = 2009}\ (section))\ -$$
$$\Pi_{course\_id}\ (\sigma_{semester = \text{“Spring”} \wedge year = 2010}\ (section))$$

  - Cartesian-Product

$$\sigma_{dept\_name = \text{“Physics”}}(instructor\ \times\ teaches)$$

  - Set-Intersection

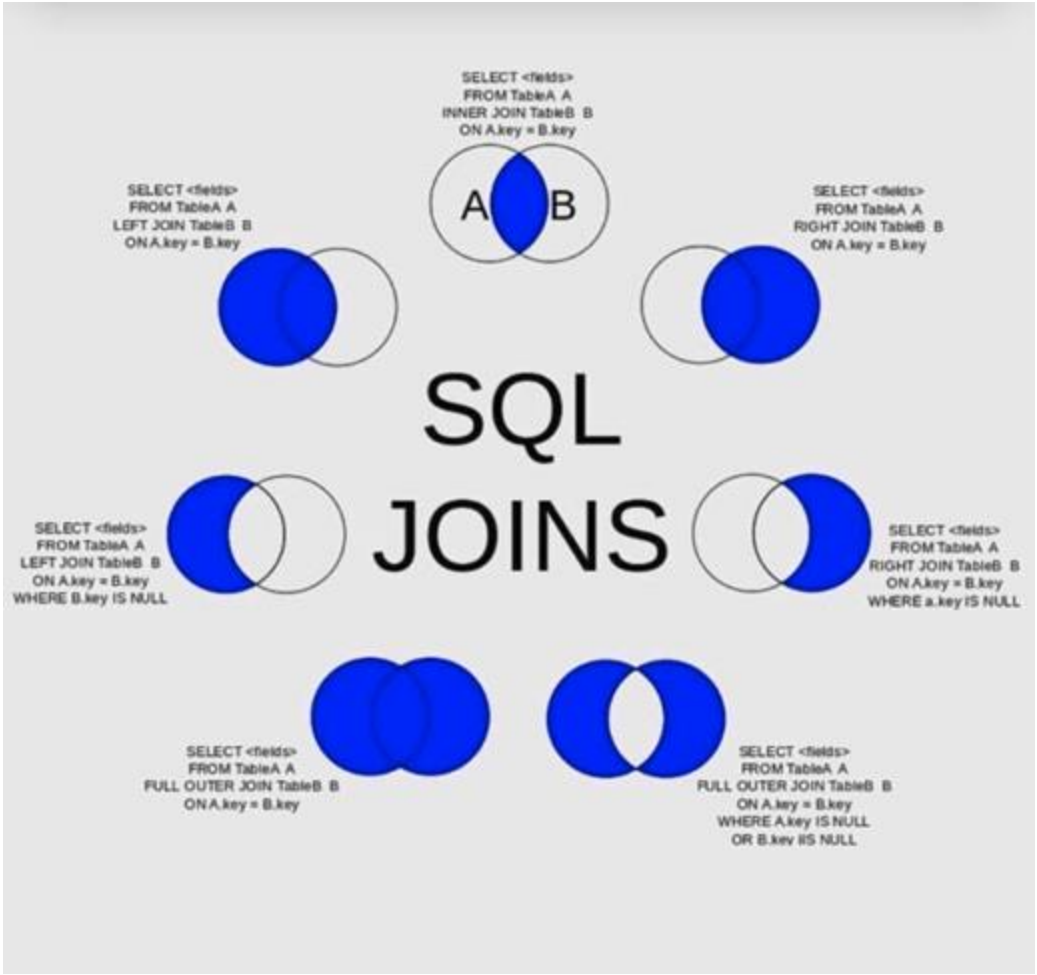
$$\Pi_{course\_id}\ (\sigma_{semester = \text{“Fall”} \wedge year = 2009}\ (section)) \cap$$
$$\Pi_{course\_id}\ (\sigma_{semester = \text{“Spring”} \wedge year = 2010}\ (section))$$

Lenguajes Formales de Consulta

- Operaciones
  - Natural-Join

$$\Pi_{name,\ course\_id}\ (instructor \bowtie teaches)$$

  - Full Outer join
  - Left Outer join
  - Right Outer join


SELECT <fields>
FROM TableA A
INNER JOIN TableB B
ON A.key = B.key
A
B
SELECT <fields>
FROM TableA A
LEFT JOIN TableB B
ON A.key = B.key
SELECT <fields>
FROM TableA A
RIGHT JOIN TableB B
ON A.key = B.key
SQL
JOINS
SELECT <fields>
FROM TableA A
LEFT JOIN TableB B
ON A.key = B.key
WHERE B.key IS NULL
SELECT <fields>
FROM TableA A
RIGHT JOIN TableB B
ON A.key = B.key
WHERE a.key IS NULL
SELECT <fields>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.key = B.key
SELECT <fields>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.key = B.key
WHERE A.key IS NULL
OR B.kev IIS NULL

Lenguajes Formales de Consulta

- Operaciones
  - Aggregate Functions
    - Sum

$$\mathcal{G}_{\mathbf{sum}(salary)}(instructor)$$

    - Count-distinct

    - Average

$$\mathcal{G}_{\mathbf{average}(salary)}(instructor)$$

Introducción a SQL

Introducción a SQL

Introducción a SQL

- DDL
  - Especifica la estructura de una base de datos
  - Define
    - Esquema de cada tabla o relación
    - Los tipos de valores de cada atributo
    - Las restricciones de integridad
    - Los índices de cada relación
    - La seguridad para el acceso
    - El almacenamiento físico

Introducción a SQL

- Tipos de datos comunes
  - char(n)
  - varchar(n)
  - text
  - int
  - numeric(p,d)
  - real
  - float(n)
  - date
  - time
  - datetime
  - timestamp

Introducción a SQL

- Otros tipos de datos
  - CLOB (Character Large OBject)
    - 10 KB
  - BLOB (Binary Large OBject)
    - Image 10MB
    - Movie 2GB
  - Definición de tipos

```
create type Dollars as numeric(12,2) final;

create table department
    (dept_name      varchar (20),
     building       varchar (15),
     budget         Dollars);
```

Introducción a SQL

- DDL
  - CREATE

create table r
$(A_1 \quad D_1,$
$A_2 \quad D_2,$
$\ldots,$
$A_n \quad D_n,$
$\langle$integrity-constraint$_1\rangle,$
$\ldots,$
$\langle$integrity-constraint$_k\rangle);$

create table *department*
(*dept_name* **varchar** (20),
building **varchar** (15),
budget **numeric** (12,2),
primary key (*dept_name*));

Introducción a SQL

- DDL
  - CREATE
    - Llave primaria
      - Identifican de manera única un registro de la tabla
      - Debe ser not null y única
    - Llave foránea
      - Identifican de manera única un registro de otra tabla
      - Normalmente es not null

Introducción a SQL

- DDL
  - CREATE

```
create table department
    (dept_name      varchar (20),
     building       varchar (15),
     budget         numeric (12,2),
     primary key (dept_name));

create table course
    (course_id      varchar (7),
     title          varchar (50),
     dept_name      varchar (20),
     credits        numeric (2,0),
     primary key (course_id),
     foreign key (dept_name) references department);
```

```
create table instructor
    (ID             varchar (5),
     name           varchar (20) not null,
     dept_name      varchar (20),
     salary         numeric (8,2),
     primary key (ID),
     foreign key (dept_name) references department);

create table section
    (course_id      varchar (8),
     sec_id         varchar (8),
     semester       varchar (6),
     year           numeric (4,0),
     building       varchar (15),
     room_number    varchar (7),
     time_slot_id   varchar (4),
     primary key (course_id, sec_id, semester, year),
     foreign key (course_id) references course);
```

Introducción a SQL

- DDL
  - CREATE

```
create table teaches
    (ID              varchar (5),
     course_id       varchar (8),
     sec_id          varchar (8),
     semester        varchar (6),
     year            numeric (4,0),
     primary key (ID, course_id, sec_id, semester, year),
     foreign key (course_id, sec_id, semester, year) references section,
     foreign key (ID) references instructor);
```

Introducción a SQL

- DDL
  - CREATE
    - create database bd;

Introducción a SQL

- DDL
  - DROP
  - ALTER

drop table r;

alter table r **add** A D;

alter table r **drop** A;

Introducción a SQL

- DML
  - INSERT

```
insert into course
        values ('CS-437', 'Database Systems', 'Comp. Sci.', 4);

insert into course (course_id, title, dept_name, credits)
        values ('CS-437', 'Database Systems', 'Comp. Sci.', 4);

insert into course (title, course_id, credits, dept_name)
        values ('Database Systems', 'CS-437', 4, 'Comp. Sci.');
```

Introducción a SQL

- DML
  - UPDATE

```
update instructor
set salary= salary * 1.05;

update instructor
set salary = salary * 1.05
where salary < 70000;
```

Introducción a SQL

- DML
  - DELETE

```
delete from instructor
where salary between 13000 and 15000;
```

```
delete from instructor
where dept_name= 'Finance';
```

Introducción a SQL

- DML
  - SELECT

```
select name
from instructor;
```

```
select distinct dept_name
from instructor;
```

```
select ID, name, dept_name, salary * 1.1
from instructor;
```

```
select name
from instructor
where dept_name = 'Comp. Sci.' and salary > 70000;
```

Introducción a SQL

- DML
  - SELECT

```
select name, instructor.dept_name, building
from instructor, department
where instructor.dept_name= department.dept_name;
```

```
select instructor.*
from instructor, teaches
where instructor.ID= teaches.ID;
```

Introducción a SQL

- DML
  - Natural Join

```
select name, course_id
from instructor, teaches
where instructor.ID= teaches.ID;
```

Introducción a SQL

- DML
  - Rename

```
select T.name, S.course_id
from instructor as T, teaches as S
where T.ID= S.ID;
```

Introducción a SQL

- DML
  - Order

```
select name
from instructor
where dept_name = 'Physics'
order by name;
```

```
select *
from instructor
order by salary desc, name asc;
```

Introducción a SQL

- DML
  - Limit
  - Count
  - Like

```
select dept_name
from department
where building like '%Watson%';
```

Introducción a SQL

- DML
  - Where predicates

```
select name
from instructor
where salary <= 100000 and salary >= 90000;
```

```
select name
from instructor
where salary between 90000 and 100000;
```

Introducción a SQL

- DML
  - Set Operations: Union

```
select course_id
from section
where semester = 'Fall' and year= 2009;
```

```
select course_id
from section
where semester = 'Spring' and year= 2010;
```

```
(select course_id
 from section
 where semester = 'Fall' and year= 2009)
union
(select course_id
 from section
 where semester = 'Spring' and year= 2010);
```

Introducción a SQL

- DML
  - Set Operations: Intersect

```
select course_id
from section
where semester = 'Fall' and year= 2009;
```

```
select course_id
from section
where semester = 'Spring' and year= 2010;
```

```
(select course_id
 from section
 where semester = 'Fall' and year= 2009)
intersect
(select course_id
 from section
 where semester = 'Spring' and year= 2010);
```

Introducción a SQL

- DML
  - Set Operations: Except

```
select course_id
from section
where semester = 'Fall' and year= 2009;
```

```
select course_id
from section
where semester = 'Spring' and year= 2010;
```

```
(select course_id
 from section
 where semester = 'Fall' and year= 2009)
except
(select course_id
 from section
 where semester = 'Spring' and year= 2010);
```

Introducción a SQL

- DML
  - Aggregate Functions (avg, min, max, sum, count)

```
select avg (salary)
from instructor
where dept_name= 'Comp. Sci.';
```

```
select count (distinct ID)
from teaches
where semester = 'Spring' and year = 2010;
```

Introducción a SQL

- DML
  - Group by

```
select dept_name, avg (salary) as avg_salary
from instructor
group by dept_name;
```

Introducción a SQL

- DML
  - Having

```
select dept_name, avg (salary) as avg_salary
from instructor
group by dept_name
having avg (salary) > 42000;
```

Introducción a SQL

- Carga masiva de datos

```
CREATE TABLE Producto (
  idProducto int(11) NOT NULL,
  nombre varchar(45) NOT NULL,
  cantidad int(11) NOT NULL,
  precio int(11) NOT NULL,
  primary key(idProducto)
)
```

```
1,Televisor,10,1000000
2,Computador,50,1200000
3,Celular,20,800000
```

```
load data infile '/tmp/datos.csv'
into table Producto
fields terminated by ',' lines terminated by '\n'
```

SQL Intermedio

SQL Intermedio

- Join (Inner Join)

```
select *
from student, takes
where student.ID= takes.ID;
```

```
select *
from student join takes on student.ID= takes.ID;
```

SQL Intermedio

- Natural Join

```
select *
from student natural join takes;
```

- Left Outer Join

```
select *
from student natural left outer join takes;
```

- Right Outer Join

```
select *
from takes natural right outer join student;
```

SQL Intermedio

- Vistas

```
create view faculty as
select ID, name, dept_name
from instructor;

create view physics_fall_2009 as
    select course.course_id, sec_id, building, room_number
    from course, section
    where course.course_id = section.course_id
                and course.dept_name = 'Physics'
                and section.semester = 'Fall'
                and section.year = '2009';
```

SQL Intermedio

- Vistas
  - Consultas

```
select course_id
from physics_fall_2009
where building= 'Watson';
```

SQL Intermedio

- DML
  - Subconsultas anidadas

```
select distinct course_id
from section
where semester = 'Fall' and year= 2009 and
        course_id in (select course_id
                      from section
                      where semester = 'Spring' and year= 2010);
```

SQL Intermedio

- DML
  - Subconsultas anidadas

```
select name
from instructor
where salary > some (select salary
                     from instructor
                     where dept_name = 'Biology');
```

```
select name
from instructor
where salary > all (select salary
                    from instructor
                    where dept_name = 'Biology');
```

SQL Intermedio

- DML
  - Subconsultas anidadas

```
select course_id
from section as S
where semester = 'Fall' and year= 2009 and
        exists (select *
                from section as T
                where semester = 'Spring' and year= 2010 and
                        S.course_id= T.course_id);
```

SQL Intermedio

- DML
  - Subconsultas anidadas

```
select distinct S.ID, S.name
from student as S
where not exists ((select course_id
                   from course
                   where dept_name = 'Biology')
                   except
                   (select T.course_id
                   from takes as T
                   where S.ID = T.ID));
```

SQL Intermedio

- DML
  - Subconsultas anidadas

```
select T.course_id
from course as T
where unique (select R.course_id
                from section as R
                where T.course_id= R.course_id and
                      R.year = 2009);
```

SQL Intermedio

- DML
  - Subconsultas anidadas

```
select T.course_id
from course as T
where not unique (select R.course_id
                  from section as R
                  where T.course_id= R.course_id and
                        R.year = 2009);
```

SQL Intermedio

- DML
  - Subconsultas anidadas

```
select dept_name, avg_salary
from (select dept_name, avg (salary) as avg_salary
        from instructor
        group by dept_name)
where avg_salary > 42000;
```

SQL Intermedio

- DML
  - Subconsultas anidadas

```
with max_budget (value) as
        (select max(budget)
        from department)
select budget
from department, max_budget
where department.budget = max_budget.value;
```