Modelo Relacional

Conceptos Básicos

Introducción a las Bases de Datos

1 Modelo de Datos

- Un **Modelo de Datos** es un lenguaje para expresar :
  - **Esquemas (Estructuras de datos):**
    Son elementos de los problemas.
    Ej.: *CURSOS(nro_curso, nombre, horas)*
  - **Restricciones:**
    Son condiciones que describen qué estructuras son válidas y cuáles no.
    Describen qué eglas que deben cumplir los datos para que la base sea considerada válida.
    Ej.: $\forall c \in Cursos.c.horas < 120$
  - **Operaciones:**
    Permiten agregar, borrar y consultar instancias de los esquemas.
    Ej.: *Insert into CURSOS (1911,'FBD',90);*

1

A modo de repaso, se recuerdan ciertos conceptos ya vistos en el curso.

Un **Modelo de Datos** es un lenguaje que permite describir **Estructuras de Datos** (o Esquemas), **Restricciones de Integridad** y **Operaciones**.

En una **base de datos**, en general, tiene un esquema que representa una estructura de datos y una instancia. La instancia cambia cada vez que se agrega, modifica o se elimina un dato de la base.

Las **Restricciones de Integridad** *se definen* a nivel del esquema como *condiciones que se deben aplicar sobre las instancias* de ese esquema. Las instancias que cumplan las condiciones son instancias (válidas) de la base pero las que no las cumplan, no pueden ser instancias de la base.

Las **Operaciones** tienen como objetivo modificar o recuperar datos de la instancia.

2 Modelo de Datos

En Entidad-Relación

- **Esquemas:**
  - Conjuntos de Entidades,
  - Relaciones
  - Atributos
  - Entidades Débiles
  - Agregaciones
  - Categorizaciones
- **Restricciones:**
  - Cardinalidad
  - Totalidad
  - Atributo Determinante
  - RNEs
- **Operaciones:** No Tiene.

En el **Modelo Entidad Relación** había diferentes estructuras y restricciones que permitían construir una representación bastante cercana a la realidad. Sin embargo, el modelo se orienta a describir la estructura y las instancias válidas y no tiene operaciones propiamente definidas.

3 Modelo Relacional

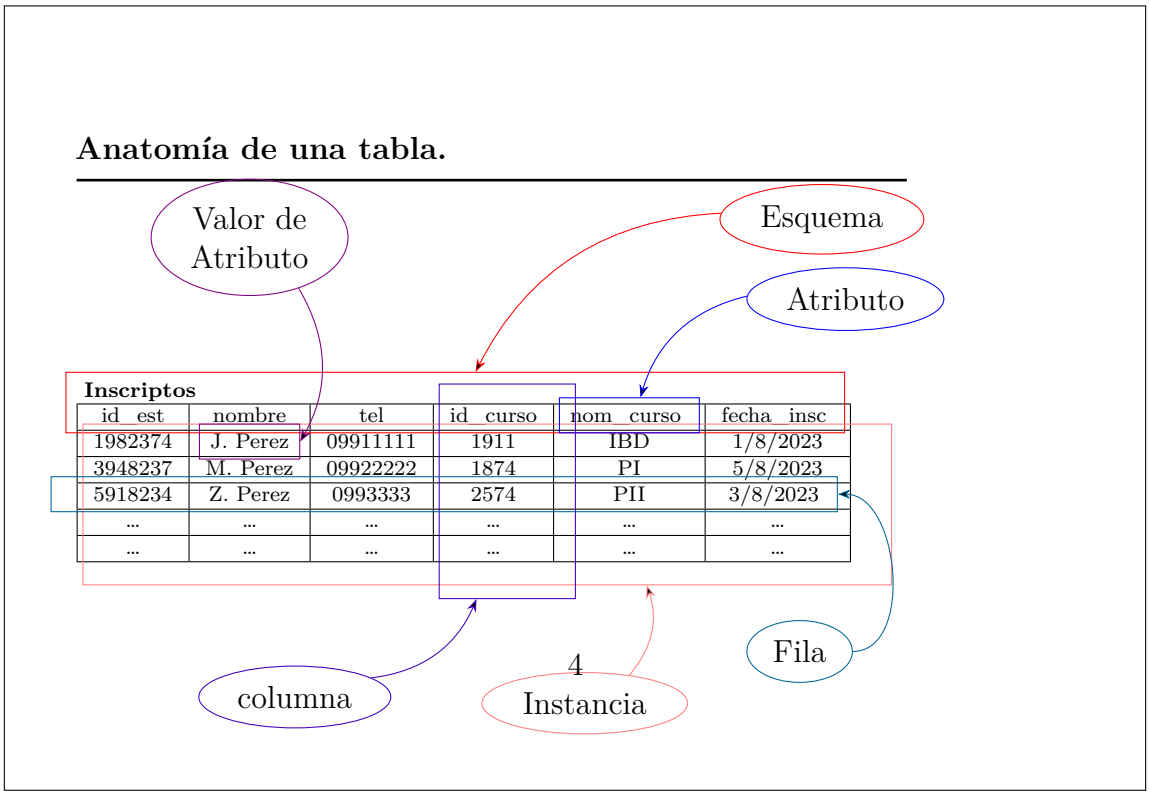
Una única estructura básica: Tabla o Relación

Inscriptos

id_est	nombre	tel	id_curso	nom_curso	fecha_insc
1982374	J. Perez	09911111	1911	IBD	1/8/2023
3948237	M. Perez	09922222	1874	PI	5/8/2023
5918234	Z. Perez	0993333	2574	PII	3/8/2023
...	...	...	...	...	...
...	...	...	...	...	...

En el **Modelo Relacional** sólo hay un único tipo de estructura: la **tabla o relación**.

4 Modelo Relacional



Una tabla, tiene un **Esquema** y una **Instancia**. El esquema, es una pareja formada por un nombre[1] y un conjunto [2] de **atributos**.

Cada **Atributo**, es un nombre con un **dominio** asociado. De esta forma, cuando se habla del atributo con nombre *nom_curso* de la tabla *Inscriptos* se sabe que sus valores sólo pueden ser strings, pero si se habla del atributo con nombre *fecha_ins* se sabe que es una fecha.

La **Instancia** de una tabla, es un conjunto de **Tuplas** que tienen que tener un valor para cada atributo que aparece en el esquema de esa tabla.

Una **columna** contiene todos los valores de la instancia para un atributo dado. Muchas veces se identifica la columna con el atributo.

Una **Tupla** es un elemento de la instancia instancia de una tabla. Esto hace que sea algo que tiene un valor para cada atributo que aparece en el esquema de la tabla. Hay dos formalizaciones para una tupla: en una, se ve como una lista de valores, en

[1] El nombre de la tabla.

[2] Podría ser una lista, dependiendo de la visión que se tome para comprender una tabla.

donde se toman los atributos en orden [3] , sin embargo en la otra, cada tupla es una función que si se le pasa un nombre de atributo, devuelve su valor.

De esta forma, la primer fila, es una función t_1 que tal que:

$$\begin{aligned} t_1(id\_est) &= 1982374 \\ t_1(nombre) &= J.Perez \\ t_1(tel) &= 0991111 \\ t_1(id\_curso) &= 1911 \\ t_1(nom\_curso) &= IBD \\ t_1(fecha\_insc) &= 1/8/2023 \end{aligned}$$

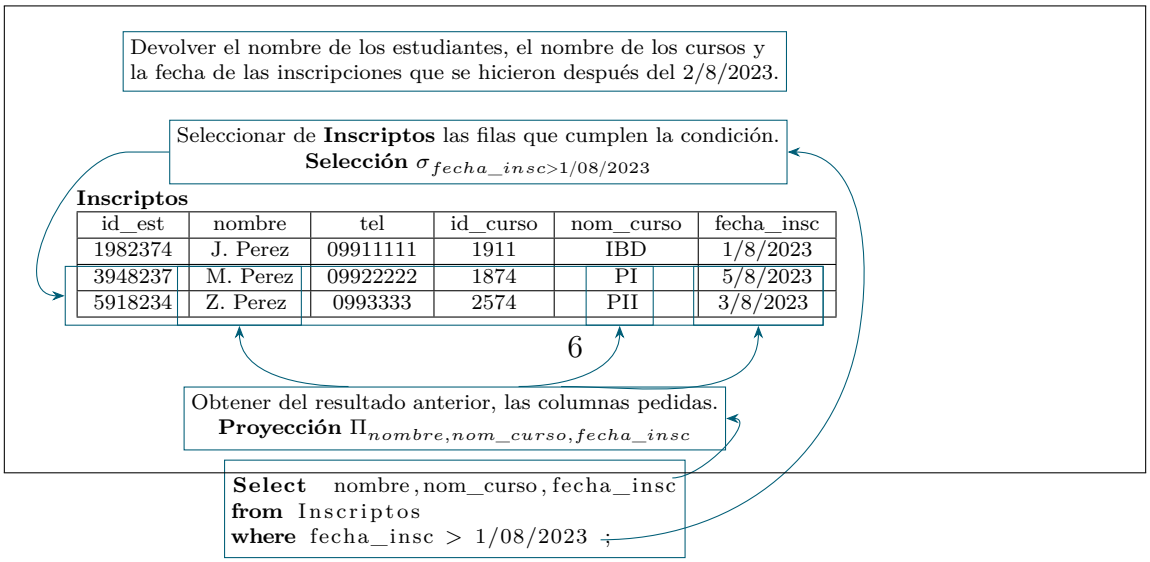
La principal ventaja de este enfoque es la posibilidad de usar nombres de atributos, de lo contrario, es necesario usar posiciones en vez de los nombres de atributos.

[3] En este caso, los atributos del esquema están en una lista lo que permite establecer un orden. En la tabla **Inscriptos** la identificación del estudiante es el primer atributo, el nombre es el segundo, etc.

5 Anatomia de una Tabla

- **Esquema:** Es una pareja que tiene un *nombre* y un conjunto de *nombres* de atributos.
- **Instancia:** Es un *conjunto de tuplas*.
- **Atributo:** Es un *nombre* que tiene un *tipo asociado*.
- **Valor de Atributo:** Es un *valor* del *tipo asociado* a un *Atributo*.
- **Tupla (Fila):** Es un elemento individual de la *Instancia* de una *Tabla* que tienen *un valor para cada atributo* del *Esquema* correspondiente.
- **Columna:** Contiene todos los *valores* de todas las *tuplas* de una *tabla* para un *atributo* dado.

6 Recuperación de datos de una tabla: Consultas



Una **Consulta** es una operación que recupera algún dato de la instancia actual de una estructura de datos.

Si piensa en términos de una única tabla, hay dos operaciones básicas: Obtener las filas que cumplen con una condición y obtener columnas que corresponden a determinados atributos. Cualquiera de estas dos operaciones devuelven un conjunto de filas, por lo que devuelven la instancia de una tabla.

El ejemplo que se ve en la imagen, es un ejemplo simple de una consulta sobre una tabla. Sin embargo, el lenguaje es mucho más pontente que y se va a presentar más adelante de una forma bastante completa.

7 Operaciones de Manipulación sobre tablas

- **Insert (Inserción):** Agrega una tupla en una tabla.

```
insert into Inscriptos
    values (1982374 , 'J. Perez', '09911111' ,
            1911 , 'IBD' , '2023-08-01');
```

- **Update (Actualización):** Actualiza uno o más atributos en una o más tuplas de una tabla.

```
update Inscriptos
    set nombre='A. Perez', tel='09911111'
    where id_est=1982374;
```

- **Delete (Borrado):** Elimina uno o más tuplas de una tabla.

```
delete from Inscriptos
where where id_est=5918234;
```

7

Además de las operaciones de consulta, podemos agregar, modificar o eliminar tuplas de las instancias de una tabla.

Para la inserción de tuplas, la instrucción básica es de la siguiente forma: **insert into ... values ....**

En el insert, las tuplas son dadas y por lo tanto no hay una especificación de tuplas.

Sin embargo, cuando se van a modificar o borrar tuplas, es necesiario dar una condición que indica cuáles son las tuplas sobre las que se quiere trabajar. Si no se pone la condición, se está trabajando sobre todas las tuplas de la tabla. Esto significa que se modifican todas las tuplas o se eliminan todas las tuplas de esa tabla.

8 Restricciones de Integridad

- Hay 3 tipos de restricciones básicas en el Modelo Relacional:
  - **Restricciones de Dominio:** Para cada atributo indica que tipo tiene.
  - **Restricciones de Identidad o Clave:** Indican que atributos de la tabla funcionan como identificadores. Esto significa que no pueden existir dos tuplas en la instancia de tabla que tengan los mismos valores en esos atributos.
  - **Restricciones de Integridad Referencial o Clave Foránea:** Para cierto conjunto de atributos de una tabla, se restringe a que sus valores deban estar en otra tabla en donde conforman una clave.

8

Hay tres tipos básicos de restricciones de integridad: las de **Dominio**, **las de Clave** y las de **Clave Foránea**.

Hay dos tipos más que viene de la visión teórica del modelo y están relaciones con las restricciones de clave y de clave foránea: la **Dependencia Funcional** y la **Dependencia de Inclusión**.

9 Restricciones de Dominio: Tipos

- Los tipos más usados para atributos son:
  - **Integer:** Enteros
  - **Float o Real:** Reales (punto flotante)
  - **Decimal o Numeric:** Permiten la definición de números de precisión arbitraria.
  - **Varchar:** Cadenas de caracteres. Se usa la comilla simple como delimitador (Ej: 'J. Perez').
  - **Date:** Fechas.
  - **Timestamp:** Fechas con horas.

9

Un **Tipo** es un conjunto de valores con un conjunto de operaciones asociado. Por ejemplo, los números enteros son un conjunto de valores que incluye a los números naturales, y los negativos de los números naturales. Las opeaciones que se pueden hacer sobre los números enteros son distintas que las que se pueden hacer sobre los números reales o sobre los strings.

Las **Restricciones de Dominio**, indican qué valores son soportados en un atributo determinado. Esto se hace indicando el **Tipo** del atributo y, posiblemente, alguna otra condición. En principio, nos ocuparemos de los tipos más usados.

Hay tres familias básicas de tipos: los tipos **Numéricos**, las **Cadenas de Caracteres o Strings** y las **Fechas**.

Los tipos numéricos son:

- **Integer:** Se usa para representar números enteros, ya sean positivos o negativos. Hay diferentes enteros de diferentes tamaños. El tamaño del entero influye en cómo se guarda internamente en la base y los mínimos y máximos que soporta. Hay, típicamente, de tres tamaños: **smallint**, **int o integer** y **bigint**.
  Los límites de los números dan una idea de lo que se puede representar:
  - Smallint: Entre -32768 y +32767. Son números relativamente chicos y en

los que los límites se pueden alcanzar fácilmente.

  - Integer: Entre -2147483648 y +2147483647. Es el tamaño que se suele utilizar cuando no hay demasiada referencia de los máximos.
  - Bigint: Entre -9223372036854775808 y +922337203685477580. En ciertas áreas de trabajo se usan bastante.

- **Float o Real:** Permite representar decimales. Tambien hay varios tamaños (Double Precision) los rangos van desde nros que tiene 6 dígitos decimales a números que tienen 15.

- **Decimal o Numeric:** Permite representar números de precisión arbitraria. La expresión *Decimal(10,4)* corresponde a los números de 10 dígitos en total de los que 4 son decimales. El primer número es la precisión y el segundo es la escala. Los límites que se soportan son realmente astronómicos. Si no se indica precisión o escala, el numero a representar pude tener 131072 digitos antes del punto decimal y 16383 después.

Para strings, hay varios tipos pero en realidad, se luele usar sólo el tipo **Varchar**. Si se indica el tamaño (ej: varchar(19)) se controla que la cantidad de caracteres sea efectivamente menor que ese valor. Esto agrega un costo de cómputo extra a la base, por lo que si no hay problemas de espacio, se suele dejar sin límites.

Por defecto, se puede guardar hasta 1GB de datos en un atributo de tipo varchar.

Hay además, varios tipos para fecha. Los dos más importantes son el tipo **Date** que soporta fechas y el tipo **Timestamp** que soporta fechas con horas e incluso, huso horario.

10 Resticciones de Clave

- Un **conjunto de atributos** de una tabla **es clave** cuando no pueden existir dos tuplas en la instancia que tengan el mismo valor para la combinación de esos atributos.
- Dada una clave de una tabla, el resto de los atributos tiene un valor único.
- Ejemplo:
  - La clave de Inscriptos es **id_est, id_curso** por lo que no pueden haber dos tuplas con *id_est = 1982374* e *id_curso = 1911*.
  - Para cada pareja *id_est, id_curso* hay un único valor posible para *nombre, tel, nom_curso, fecha_insc*.
- Una tabla puede tener varias claves.
- Las claves surgen de la realidad y no de las instancias.

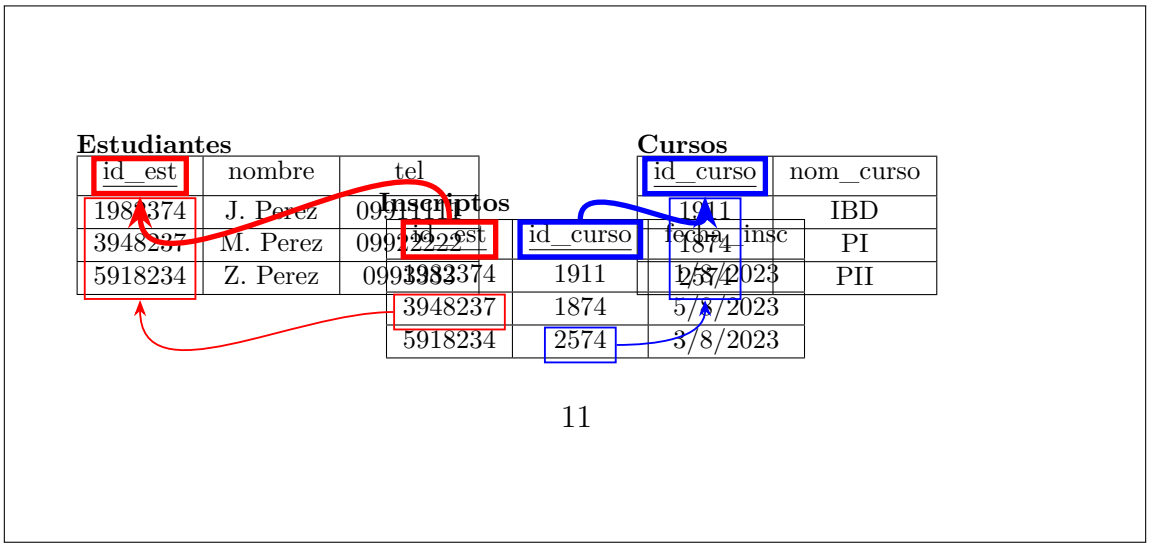
10

Las **claves** sirven para identificar las tuplas. Esto significa que una vez que se determinó que un conjunto de atributos conforman una clave de una tabla, en ninguna instancia de esa tabla pueden existir dos tuplas que tengan los mismos valores en esos atributos.

Por ejemplo, en la tabla **Inscriptos**, la clave está formada los atributos **id_est, id_curso**. Esto significa que no pueden haber dos tuplas que tengan el valor id_est = 1982374 y además el valor id_curso = 1911. Sí pueden existir dos tuplas con el mismo valor de id_est y distinto valor en id_curso o al revés.

Por otro lado, signfica también que dado un valor para la clave, el resto de los datos de la tabla son únicos.

11 Clave Foránea



Para comprender bien las restricciones de clave foránea, primero conviene representar mejor la informción la tabla de inscriptos. En la tabla, quedan datos repetidos, dado que cada vez que aparezca un estudiante inscripto en diferentes cursos hay que repetir todos los datos del estudiante. También cuando en un mismo curso aparecen diferentes estudiantes, hay que repetir todos los datos del curso.

Esto se soluciona manejando 3 tablas: Una para los datos de los estudiantes, otra para los datos del curso y otra para los datos de la inscripción.

Los atributos que están subrayados son la clave en cada tabla. En partcular, en la tabla *Inscriptos*, la clave está formada por dos atributos.

La restricción de **Clave Foránea** asegura que los valores de un conjunto de atributos en una tabla tiene que estar entre los valores de los atributos de una clave de otra tabla.

En este ejemplo, los valores e *id_est* en *inscriptos*, tienen que estar en el atributo *id_est* de *Estudiantes*.

12 Ejemplos de Creación de las Tablas

```
Create table Estudiantes (
    id_est integer
        primary key,
    nombre varchar,
    tel varchar
);
```

```
Create table Cursos (
    id_curso integer
        primary key,
    nom_curso varchar
);
```

```
Create table inscriptos (
    id_est integer  references estudiantes(id_est),
    id_curso integer,
    fecha_ins Date,
    primary key (id_est,id_curso),
    foreign key (id_curso) references curso(id_curso)
);
```

12

Tipo Clave Primaria Clave Foránea

Observar la posición de las restricciones