

Databases Midterm Study Guide

Schema: A set of attributes

Attribute: A name and type

Instance: The value in a table


Tuple: One row


Relation: Schema + Instance

A set of fields is a superkey if no two rows have the same values in those fields.

A set of fields is a key if: It is a superkey, no proper subset of its fields is a superkey.



 = entity set

 = attribute

underline = key

Diamond = relationship set “Works in” doesn’t need any attributes

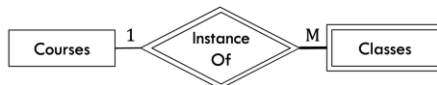
Annotate opposite edge of relationship with cardinality.

Bold line indicates if an entity must participate.

A weak entity can’t be identified by its own attributes, it’s identified by a combination of it’s own attributes and a foreign key.

•Classes is a **weak entity** set

- Double rectangle for the entity set
- Double diamond for the **supporting relationship**
- Supporting relationship must be 1-to-M, and weak entity must participate



Dashed underline for partial key.

MySQL datatypes

Int, int unsigned, Reals (floats, double, decimal), Dates, Strings char, varchar.

Entity Sets

Every entity set translates directly to a schema.

Attributes become columns

Pick one of the key attribute sets as the primary key.

Relationship Sets to Schema

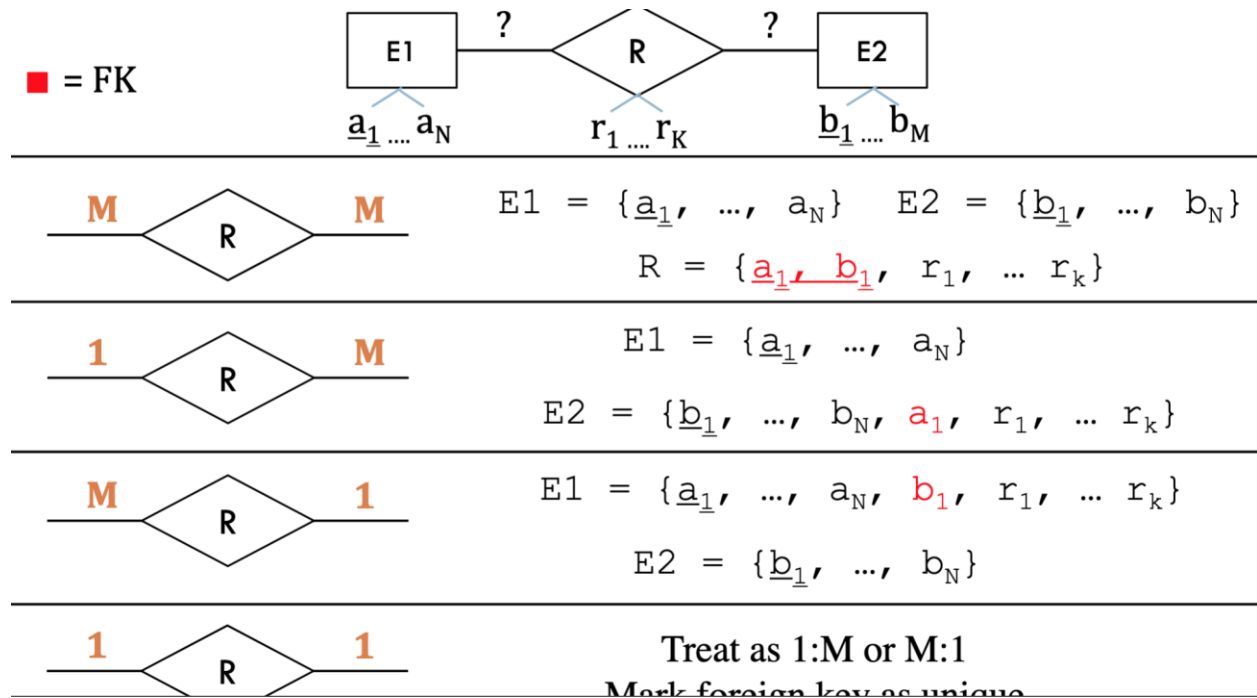
M to M – New schema for relationship

Primary keys for relating entities as foreign keys

1 to M – Merge M relationship into entity

If participation is required set not null, otherwise NULL is allowed.

1 to 1 – Treat as 1 to M, merge relationship into one of the other tables.
 If participation is required on one side, use that side.
 Add NOT NULL



Mark foreign key as unique

Weak Entity vs. Participation

Weak – A class is an instance of a specific course

Part – A department must have manager, but manager can change

i.e. a class cannot change which course it is tied to

Is-A to Schema

Every entity becomes its own schema, pull down primary key

If Base type is abstract pull down all attributes, no schema for base type

•Five basic operators

- π Projection
 - σ Selection
 - \times Cross Product
 - $-$ Set Difference
 - \cup Set Union
- where*

No duplicates in relational algebra

For union to work relations must be union compatible, same columns in same order.

Division

A		B	A / B
x	y	y	x
c1	p1	p2	c1
c1	p2	p4	c4
c1	p3		
c1	p4		
c2	p1		
c2	p2		
c3	p2		
c4	p2		
c4	p4		

Rename

Rename

- Often useful to declare 'variables' by renaming a relation
 - Not really an operator

• ρ (newname, expression)

• Example:

ρ (HerbertSerials, π Serial(σ Author=Herbert(Titles \bowtie Inventory)))

- SQL has pseudo regex:

... WHERE Title LIKE 'J_ %m'

- '_' means any one character
- '%' means 0 or more arbitrary characters

Some example Adv. Queries

- Find all students younger than everyone in 'Databases'

```
select s.sName from Students s
where s.DOB > all
  (select s2.DOB from Students s2
   natural join Enrolled natural join
   Courses c where c.cName='Databases');
```

•Find students taking all classes:

```
select s.sName
from Students s
where not exists
```

which makes
this true

```
(select c.cID
from Courses c
where not exists
```

If inner select is non-empty,
then this is false

If false for all {s, c},
then this select is empty

foreach Student s

foreach Course c

foreach Enrollment e

```
select e.cID
from Enrolled e
where e.cID = c.cID
and e.sID = s.sID));
```

If there is an enrollment for
{s, c}

Then this select is non-empty

•Useful in combination with ORDER BY

- Find ... with smallest CardNum

```
SELECT ... LIMIT 1 ORDER BY CardNum;
```

- Find ... with biggest CardNum

```
SELECT ... LIMIT 1 ORDER BY CardNum DESC;
```

Aggregates

Count()

Max()

Min()

Sum()

Avg()

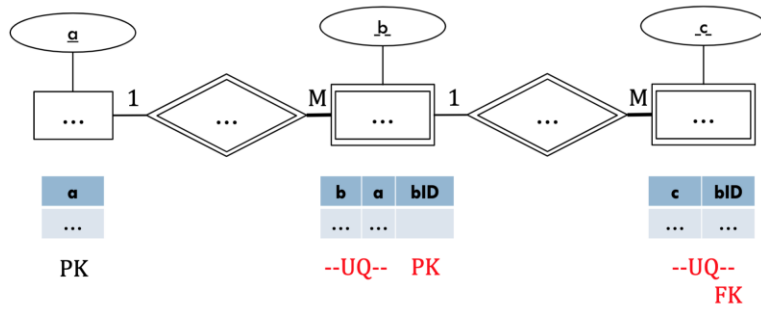
•How many copies of 'Harry Potter'?

```
SELECT COUNT(*)
FROM Titles NATURAL JOIN Inventory
WHERE Title='Harry Potter';
```

Group by – gives one representative row for each group in the specified column

Weak Entity Chain to Schema

- Add a new PK to represent each {b, a}



- bID now represents complex key in smaller footprint