# Keys and Null Values



If there are nulls, keys do not work that well:

- They do not guarantee unique identification;
- They do not help in establishing correspondences between data in different relations

| RegNum | Surname | FirstName | BirthDate | DegreeProg |
|---|---|---|---|---|
| NULL | Smith | John | NULL | Computing |
| 587614 | Smith | Lucy | 01/05/61 | Engineering |
| 934856 | Black | Lucy | NULL | NULL |
| NULL | Black | Lucy | 05/03/58 | Engineering |

- Are the third and fourth tuple the same?
- How do we access the first tuple?

# Primary Keys



- The presence of nulls in keys has to be limited.
- Each relation must have a *primary key* on which nulls are not allowed (in any attribute)
- Notation: the attributes of the primary key are *underlined*
- References between relations are realized through primary keys

| RegNum | Surname | FirstName | BirthDate | DegreeProg |
|---|---|---|---|---|
| 643976 | Smith | John | NULL | Computing |
| 587614 | Smith | Lucy | 01/05/61 | Engineering |
| 934856 | Black | Lucy | NULL | NULL |
| 735591 | Black | Lucy | 05/03/58 | Engineering |

# Do we Always Have Primary Keys?



- In most cases, we do have reasonable primary keys (e.g., student number, SIN)
- There may be multiple keys, one of which is designated as primary.

# Recap



- A set of fields is a *key* for a relation if:
  1. No two distinct tuples can have same values in all key fields, and
  2. This is not true for any subset of the key.
- If #2 false, then a *superkey*.
- If there's >1 key for a relation, one of the keys is chosen to be the *primary key*.
- E.g., *sid* is a key for Students. (What about *name*?) The set {*sid, gpa*} is a superkey.

# Primary and Candidate Keys



**Enrolled(sid, cid, grade)**

1. "For a given student and course, there is a single grade." vs.
2. "Students can take only one course, and receive a single grade for that course; further, no two students in a course receive the same grade."

**Enrolled(<u>sid, cid</u>, grade)**

**Enrolled(<u>sid</u>, cid, grade)**

- key (cid, grade)

- Be careful to define Integrity Constraints (ICs) correctly at design time.
- ICS are checked when data is updated.

# Foreign Keys



- Pieces of data in different relations are correlated by means of values of primary keys.
- Referential integrity constraints are imposed in order to guarantee that the values refer to existing tuples in the referenced relation.
- A *foreign key* requires that the values on a set X of attributes of a relation $R_1$ must appear as values for the primary key of another relation $R_2$.
  - In other words, set of attributes in one relation that is used to `refer' to a tuple in another relation. (Must correspond to primary key of the second relation.) Like a `logical pointer'.

# Referential Integrity



- E.g. *sid* is a foreign key referring to Students:
  - Enrolled(*sid*: string, *cid*: string, *grade*: string)
  - If all foreign key constraints are enforced, *referential integrity* is achieved, i.e., no dangling references.

# Referential Integrity (cont'd)



- Only students listed in the Students relation should be allowed to enroll for courses.

Enrolled

| sid | cid | grade |
|---|---|---|
| 53666 | Carnatic101 | C |
| 53666 | Reggae203 | B |
| 53650 | Topology112 | A |
| 53666 | History105 | B |

Students

| sid | name | login | age | gpa |
|---|---|---|---|---|
| 53666 | Jones | jones@cs | 18 | 3.4 |
| 53688 | Smith | smith@eecs | 18 | 3.2 |
| 53650 | Smith | smith@math | 19 | 3.8 |