Cheat sheet

select: σpredicate select the tuple that meets predicate condition

project: ∏attribute1, attribute2.. select the attribute columns

union: ∪ take the union of tuples of two relations having the same attribute

set difference:   A-B remove the available tuple in B from A Cartesian product: AxB put all the possible tuple combination of A and B

rename: ρ

Domain type: varchar (maxSize), char(fixedN), int, smallint, numeric(p,d), float(n) precision at least n digits create table r (A1 D1, A2 D2, ..., An Dn, ) (integrity‐constraint: not null, primary key, foreign key references r1),

add tuple : insert into table values ( a1,a2,…an)

drop table table allows remove a relation;

like '% substring %' % character matches any substring .  set operator union, intersect, except (equivalent to set difference)   f

alter table add AD allow adding more attribute, alter table drop D removes  attribute

select A1, A2, ..., An , *allows selection of tuple of attributes A1, A2, ..., An*

from r1, r2, ..., rm*, the cartisean  relation r1, r2, ..., rm*

where P *: where predicate (condition) P is true*

group by allow aggregate function (avg, max, min, sum, count)  not in R return / update tuple not in R  with clause provi where predicate (condition) P is true  des a way of defining a temporary view whose definition is available only to the query in which the with  clause occurs

Join operations take two relations and return as a result another relation as a join on (using) a common attribute(s).

**Rule 1**) Strong entity E:  Create relation with attributes of E , Primary key is equal to the PK of E **Rule 2**) Weak entity W identified by E through relationship R: Create relation with attributes of W and R and PK(E). Set PK  to discriminator attributes combined with PK(E). PK(E) is a foreign key to E.

***Rule 3)*** Binary relationship R between A and B: one‐to‐one: If no side is total add PK of A to as foreign key in B or the other way around. Add any attributes of the relationship R to A respective B. If one side is total add PK of the other‐side as foreign key. Add any attributes of the relationship R to the total side. If both sides are total merge the two relation into

a new relation E and choose either PK(A) as PK(B) as the new PK. Add any attributes of the relationship R to the new relation E.

***Rule 4)*** Binary relationship R between A and B: one‐to‐many/many‐to‐one. Add PK of the “one” side as foreign key to the

“many” side. Add any attributes of the relationship R to the “many” side.

***Rule 5)*** Binary relationship R between A and B: many‐to‐many: Create a new relation R. Add PK’s of A and B as attributes

+ plus all attributes of R. The primary key of the relationship is PK(A) + PK(B). The PK attributes of A/B form a foreign key *to A/B*

***Rule 6)*** N‐ary relationship R between E1 … En. Create a new relation. Add all the PK’s of E1 … En. Add all attributes of R to

the new relation. The primary key or R is PK(E1) … PK(En). Each PK(Ei) is a foreign key to the corresponding relation.

***Rule 7)*** Entity E with multi‐valued attribute A. Create new relation. Add A and PK(E) as attributes. PK is all attributes.

PK(E) is a foreign key.