Week8_Handout

Database System-SQL Advanced

Database System-SQL Advanced-part 1

More SQL

- Database Modification
- Assertions and Triggers
- Procedures Stored in the Database
- Views, Index
- Privileges, Grant and Revoke
- Grant Diagrams

DB Modifications

- *Modification* = insert, delete, or update.

Syntax

- INSERT INTO relation VALUES (list of values);
- INSERT INTO relation (subquery);
- DELETE FROM relation WHERE condition;
- UPDATE relation SET assignments
  WHERE condition;

Example

Likes(drinker, beer)

- Insert the fact that Sally likes Bud.

```
INSERT INTO Likes(drinker, beer)
        VALUES('Sally', 'Bud');
```

Insertion of the Result of a Query

INSERT INTO relation (subquery);

Example Frequents(drinker, bar)

- Create a table of all Sally's potential buddies, i.e., the people who frequent bars that Sally also frequents.

```
CREATE TABLE PotBuddies(name char(30));

INSERT INTO PotBuddies
(SELECT    DISTINCT d2.drinker
 FROM      Frequents d1, Frequents d2
 WHERE     d1.drinker = 'Sally' AND
                d2.drinker <> 'Sally' AND
                d1.bar = d2.bar);
```

Deletion

DELETE FROM relation WHERE condition;

– Deletes all tuples satisfying the condition from the named relation.

Example Likes(drinker, beer)

- Sally no longer likes Bud.

```
DELETE FROM  Likes
WHERE  drinker = 'Sally' AND beer = 'Bud';
```

- Make the Likes relation empty.

```
DELETE FROM Likes;
```

Different from DROP TABLE Likes;

Updates

UPDATE relation SET assignments WHERE condition;

Example Drinkers(name, addr, phone)

- **Drinker Fred's phone number is 555-1212.**

```
UPDATE    Drinkers
SET       phone = '555-1212'
WHERE     name = 'Fred';
```

Example Sells(bar, beer, price)

- **Make $4 the maximum price for beer.**

```
UPDATE    Sells
SET       price = 4.00
WHERE     price > 4.00;
```

SQL Assertions

- Database-schema constraint.
- Not present in MySQL.
- Keep always true.
- Checked whenever a mentioned relation changes. Change may be rejected.
- Syntax:

```
CREATE ASSERTION <name>
CHECK  (<condition>);
DROP ASSERTION <name>
```

Example

Sells(bar, beer, price)

- No bar may charge an avg. of more than $5 for beer.

```
CREATE ASSERTION NoRipoffBars
CHECK (NOT EXISTS(
        SELECT bar
        FROM Sells
        GROUP BY bar
        HAVING 5.0 < AVG(price)));
```

```
CHECK (5.0>= ALL
(SELECT AVG(price)
FROM Sells
GROUP BY bar))
```

- Checked whenever Sells changes.

Example

Bars(name, addr)
Drinkers(name, addr, phone)

- There cannot be more bars than drinkers.

```
CREATE ASSERTION FewBar
CHECK( (SELECT COUNT(*) FROM Bars) <=
                    (SELECT COUNT(*) FROM Drinkers));
```

- Checked whenever Bars or Drinkers changes.

Triggers

Often called event-condition-action rules.

- *Event* = a class of changes in the DB, *e.g.*, insertion, deletion or update.
- *Condition* = a test as in a where-clause for whether or not the trigger applies.
- *Action* = one or more SQL statements.
- Triggers vs Assertions:
  - Assertion is a boolean-valued SQL expression that must be true at all times.
  - Triggers have actions associated that must be performed whenever the trigger event arises.

Example （MySQL version）

Sells(bar, beer, price); Beers(name, manf)

- Whenever a new tuple is inserted into Sells:
  - If the beer mentioned is not in Beers, then insert it (with a null manufacturer).

```
delimiter //
CREATE TRIGGER BeerTrig
BEFORE INSERT ON Sells FOR EACH ROW
BEGIN
IF NEW.beer not in (select name from beers) THEN
INSERT INTO Beers(name) values(NEW.beer);
END IF;
END; //
delimiter ;
```

https://dev.mysql.com/doc/refman/8.0/en/trigger-syntax.html

Also, OLD.beer in case of delete.

Some Options

- BEFORE can be AFTER.
- INSERT can be DELETE or UPDATE OF <attr>.
- Condition is optional.

Example

Sells(bar, beer, price); RipoffBars(bar)

- Maintain a list of all the bars that raise their price for some beer by more than $1.

```
delimiter //
create trigger PriceTrig
AFTER UPDATE ON sells
FOR EACH ROW
BEGIN
IF NEW.price > OLD.price+1 THEN
INSERT INTO RipoffBars values (new.bar);
END IF;
END; //
delimiter ;
```

Example

Employee

empno	empname	deptno
2581	徐唱	d2
9031	李静	d2
10102	王闻刚	d3
18316	冯新	d1
25348	张风	d3
28559	刘国风	d1
29346	赵东生	d2

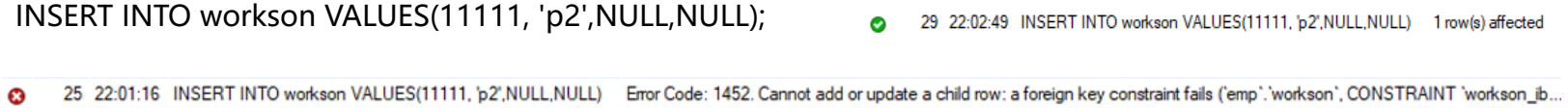
Workson

empno	projectno	job	enterdate
2581	p3	分析员	98-10-15
9031	p1	管理员	98-4-15
9031	p3	职员	97-11-15
10102	p1	分析员	97-1-10
10102	p3	管理员	99-1-1
18316	p2	职员	98-2-15
25348	p2	<NULL>	98-6-1
28559	p1	<NULL>	98-8-1
28559	p2	职员	99-2-1
29346	p1	职员	98-1-4
29346	p2	<NULL>	97-12-15

```
delimiter //
CREATE TRIGGER WorksonInsertTrig
BEFORE INSERT ON workson FOR EACH ROW
BEGIN
IF NEW.empno not in (select empno from employee) THEN
INSERT INTO employee(empno) values(NEW.empno);
END IF;
END; //
delimiter ;
```

将职员号码1111插入表workson的新行中

INSERT INTO workson VALUES(11111, 'p2',NULL,NULL);



Example

Employee

empno	empname	deptno
2581	徐唱	d2
9031	李静	d2
10102	王闻刚	d3
18316	冯新	d1
25348	张风	d3
28559	刘国风	d1
29346	赵东生	d2

Workson

empno	projectno	job	enterdate
2581	p3	分析员	98-10-15
9031	p1	管理员	98-4-15
9031	p3	职员	97-11-15
10102	p1	分析员	97-1-10
10102	p3	管理员	99-1-1
18316	p2	职员	98-2-15
25348	p2	<NULL>	98-6-1
28559	p1	<NULL>	98-8-1
28559	p2	职员	99-2-1
29346	p1	职员	98-1-4
29346	p2	<NULL>	97-12-15

```
delimiter //
CREATE TRIGGER EmployeeDeleteTrig
Before delete on employee for each row
begin
if old.empno in (select empno from workson) then
delete from workson where empno=old.empno;
end if;
end; //
delimiter ;
```

删除职员号为9031的行

DELETE FROM employee WHERE empno=9031;

```
49  22:27:40  DELETE FROM employee WHERE empno=9031   1 row(s) affected
```

```
44  22:23:19  DELETE FROM employee WHERE empno=9031   Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails (`emp`.`workson`, CONSTRAINT `workso..
```

Summary

- Database Modification
- Assertions
- Triggers

多选题 1分

互动交流——不定项选择

向关系Department(deptno,deptname,location)插入信息：宣传部d4在上海，正确的SQL语句是

A INSERT INTO Department Values('d4','宣传部','上海');

B INSERT INTO Department(deptno,deptname,location) Values('d4','宣传部','上海');

C INSERT INTO Department(location,deptname,deptno) Values('上海','宣传部','d4');

D INSERT INTO Department Values('上海','宣传部','d4');

提交

主观题 10分

互动交流二

刘国风不再参与网络布线项目

Employee

empno	empname	deptno
2581	徐唱	d2
9031	李静	d2
10102	王闻刚	d3
18316	冯新	d1
25348	张风	d3
28559	刘国风	d1
29346	赵东生	d2

Project

projectno	projectname	budget
p1	网络布线	120000
p2	软件升级	95000
p3	系统开发	185600

Workson

empno	projectno	job	enterdate
2581	p3	分析员	98-10-15
9031	p1	管理员	98-4-15
9031	p3	职员	97-11-15
10102	p1	分析员	97-1-10
10102	p3	管理员	99-1-1
18316	p2	职员	98-2-15
25348	p2	<NULL>	98-6-1
28559	p1	<NULL>	98-8-1
28559	p2	职员	99-2-1
29346	p1	职员	98-1-4
29346	p2	<NULL>	97-12-15

提交

讲解

Employee

empno	empname	deptno
2581	徐唱	d2
9031	李静	d2
10102	王闻刚	d3
18316	冯新	d1
25348	张风	d3
28559	刘国风	d1
29346	赵东生	d2

Project

projectno	projectname	budget
p1	网络布线	120000
p2	软件升级	95000
p3	系统开发	185600

Workson

empno	projectno	job	enterdate
2581	p3	分析员	98-10-15
9031	p1	管理员	98-4-15
9031	p3	职员	97-11-15
10102	p1	分析员	97-1-10
10102	p3	管理员	99-1-1
18316	p2	职员	98-2-15
25348	p2	<NULL>	98-6-1
28559	p1	<NULL>	98-8-1
28559	p2	职员	99-2-1
29346	p1	职员	98-1-4
29346	p2	<NULL>	97-12-15

刘国风不再参与网络布线项目

```
delete from workson
where empno=(select empno from employee where empname='刘国风')
and projectno= (select projectno from project where projectname='网络布线');
```

主观题 10分

互动交流三

写出 对于样本数据库表project,将所有项目的预算增加15000 的SQL语句

Project

projectno	projectname	budget
p1	网络布线	120000
p2	软件升级	95000
p3	系统开发	185600

提交

讲解

Project

projectno	projectname	budget
p1	网络布线	120000
p2	软件升级	95000
p3	系统开发	185600

```
update project set
budget=budget+15000;
```

select* from project

projectno	projectname	budget
p1	网络布线	135000
p2	软件升级	110000
p3	系统开发	200600

主观题 10分

互动交流四

创建触发器，使得如下SQL语句可以正确执行：

UPDATE workson SET empno=11111 WHERE empno=9031;

Employee

empno	empname	deptno
2581	徐唱	d2
9031	李静	d2
10102	王闻刚	d3
18316	冯新	d1
25348	张风	d3
28559	刘国风	d1
29346	赵东生	d2

Workson

empno	projectno	job	enterdate
2581	p3	分析员	98-10-15
9031	p1	管理员	98-4-15
9031	p3	职员	97-11-15
10102	p1	分析员	97-1-10
10102	p3	管理员	99-1-1
18316	p2	职员	98-2-15
25348	p2	<NULL>	98-6-1
28559	p1	<NULL>	98-8-1
28559	p2	职员	99-2-1
29346	p1	职员	98-1-4
29346	p2	<NULL>	97-12-15

提交

讲解

创建触发器，使得如下SQL语句可以正确执行：

UPDATE workson SET empno=11111 WHERE empno=9031;

Employee

empno	empname	deptno
2581	徐唱	d2
9031	李静	d2
10102	王闻刚	d3
18316	冯新	d1
25348	张风	d3
28559	刘国风	d1
29346	赵东生	d2

Workson

empno	projectno	job	enterdate
2581	p3	分析员	98-10-15
9031	p1	管理员	98-4-15
9031	p3	职员	97-11-15
10102	p1	分析员	97-1-10
10102	p3	管理员	99-1-1
18316	p2	职员	98-2-15
25348	p2	<NULL>	98-6-1
28559	p1	<NULL>	98-8-1
28559	p2	职员	99-2-1
29346	p1	职员	98-1-4
29346	p2	<NULL>	97-12-15

```
delimiter //
create trigger WorksonUpdateTrig
BEFORE UPDATE ON workson FOR EACH ROW
BEGIN
IF NEW.empno NOT IN (SELECT empno From employee) THEN
INSERT INTO employee(empno) values (new.empno);
END IF;
END; //
delimiter ;
```

主观题 10分

互动交流五

是否可以使用触发器支持如下操作？如果能，简述如何设计触发器。如果触发器无法实现，是否能用其他方式支持如下SQL语句正确执行?

Employee

empno	empname	deptno
2581	徐唱	d2
9031	李静	d2
10102	王闻刚	d3
18316	冯新	d1
25348	张风	d3
28559	刘国风	d1
29346	赵东生	d2

Workson

empno	projectno	job	enterdate
2581	p3	分析员	98-10-15
9031	p1	管理员	98-4-15
9031	p3	职员	97-11-15
10102	p1	分析员	97-1-10
10102	p3	管理员	99-1-1
18316	p2	职员	98-2-15
25348	p2	<NULL>	98-6-1
28559	p1	<NULL>	98-8-1
28559	p2	职员	99-2-1
29346	p1	职员	98-1-4
29346	p2	<NULL>	97-12-15

修改表employee中的职员号为9031相对应的行

```
UPDATE employee
SET empno=22222
WHERE empno=9031;
```

提交

Database System-SQL Advanced-part 2

Stored Routine

- A stored routine is either a procedure or a function.
- Created with the CREATE PROCEDURE and CREATE FUNCTION statements.
- A procedure is invoked using a CALL statement.
- A function can be called from inside a statement just like any other function.

Stored Procedure

```
CREATE TABLE print(printInfo char(50));
delimiter \\
CREATE PROCEDURE printProc(inout printInfo char(50))
#CREATE PROCEDURE printProc()
BEGIN
#CREATE TABLE print(printInfo char(50));
IF (SELECT count(*)
      FROM workson
      WHERE projectno='p1')>3 THEN
     set printInfo='The number in the project p1 is 4 or more';
ELSE  set printInfo ='The number in the project p1 is less than 4';
END IF;
END; \\
delimiter ;
CALL printProc(@a);
SELECT @a;
```

	@a
▶	The number in the project p1 is 4 or more

Stored Function

```
CREATE FUNCTION hello (s CHAR(20))
RETURNS CHAR(50) DETERMINISTIC
RETURN CONCAT('Hello, ',s,'!');
SELECT hello('world');
```

	hello('world')
▶	Hello, world!

Local variable

- Local variables can be declared within stored programs .
- To provide a default value for a variable, include a DEFAULT clause.
- If the DEFAULT clause is missing, the initial value is NULL.

Stored Procedure

```
delimiter \\
create procedure printBudget(out p_out char(50))
BEGIN
DECLARE avg_budget int;
DECLARE extra_budget int;
SET extra_budget=15000;
SELECT AVG(budget) INTO avg_budget FROM project;
IF(SELECT budget FROM project WHERE projectno='p1')<avg_budget THEN
UPDATE project
SET budget=budget+extra_budget
WHERE projectno='p1';
SET p_out='Budget for p1 increased by 15000';
ELSE set p_out='budget for p1 unchanged';
END IF;
END; \\
delimiter ;
set @p_out='';
call printBudget(@p_out);
select(@p_out);
```

(@p_out)
Budget for p1 increased by 15000

触发器举例

```
create table audit_budget
      (projectno char(4) null,
       username char(16) null,
       time datetime null,
       budgetold float null,
       budgetnew float null)
```

```
delimiter \\
create trigger modify_budget
after update on project
for each row
   BEGIN
      DECLARE budget_old FLOAT;
      DECLARE budget_new FLOAT;
      DECLARE project_number CHAR(4);
      set budget_old=OLD.budget;
      set budget_new=NEW.budget;
      set project_number=OLD.projectno;
       insert into audit_budget values
          (project_number,current_user(),curtime(),
          budget_old,budget_new);
   END; \\
delimiter ;
```

```
update project
set budget=150000
where projectno='p1';
select * from audit_budget;
```

	projectno	username	time	budgetold	budgetnew
▶	p1	root@localhost	2020-02-22 02:05:47	120000	150000

嵌入式 SQL(embedded SQL)

- SQL语言实现复杂应用有它的不足
  - 查询来计算一个数n的阶乘
  - SQL不能把它的输出直接转化为图形形式
- 实际的数据库编程同时需要SQL和传统语言
  - 在高级语言中出现的SQL命令称为嵌入式SQL
  - 嵌入SQL的高级语言称为宿主语言(host language)
  - 例如，C#通过ODBC接口进行数据库访问，JAVA通过JDBC接口进行数据库访问

建立视图

视图是从一个或几个基本表（或视图）导出的虚拟表，
　可以简化用户查询操作，对数据提供安全保护
语句格式：CREATE VIEW 视图名[(视图列名表)]

AS {SELECT 语句}

[WITH CHECK OPTION]

- SELECT语句可以是任意复杂的SELECT语句，但通常不允许含有ORDER BY子句
- 视图列名表或者全部省略或者全部指定
- 不能省略的情况：某列是聚集函数或表达式、包含同名列、重命名

建立视图举例

wokrson(empno,projectno,job,enterdate)

```
CREATE VIEW 职员
   AS SELECT empno,projectno,enterdate
        FROM workson
        WHERE job='职员';
CREATE VIEW v_count(projectno,countproject)
   AS SELECT projectno,COUNT(*)
        FROM workson
        GROUP BY projectno;
```

- 执行CREATE VIEW语句只是把视图的定义存入数据字典，并不执行SELECT语句
- 对视图进行查询时，才按视图的定义从基本表中查出数据

视图的查询

- 视图的查询和基本表的查询相同

```
SELECT *
FROM v_count
WHERE countproject<3;
```

删除视图

- DROP VIEW v_count
  - 只删除视图定义，并不影响导出视图的表中的任何元组
- DROP TABLE workson
  - 不仅使表workson删除，也使得视图v_count不可用

视图的更新

- 不是所有视图都允许更新（一般要求单个基本表导出）
- 转变为一个等价的对基本表的更新
- 视图的更新施加限制

```
CREATE VIEW v_1997_check
  AS SELECT empno,projectno,enterdate
       FROM workson
       WHERE enterdate BETWEEN '1997-01-01' AND '1997-12-31'
       WITH CHECK OPTION;
 INSERT INTO v_1997_check
VALUES(18316,'p1','1998-1-15')
```

索引的建立与删除

- 加快查询速度（例如：顺序查找 vs. 二叉树查找）
- CREATE INDEX 索引名 ON 表名 (列名表);

  CREATE INDEX i_empno ON workson(empno);

  CREATE INDEX i_pjno_job ON workson(projectno, job);

- DROP INDEX 索引名 ON 表名

  DROP INDEX i_empno on workson;

DBMS系统自动选择合适的索引结构（B+树、HASH索引）进行索引创建，不需要用户显式指定

索引一经创建，就由系统负责使用和维护，不需用户干预

创建索引的要点

- 每个索引要使用一定量的磁盘空间，所以索引页的数目可能会超过数据库中数据页的数目
- 与为检索使用索引的好处比，向有索引列的表进行插入、删除、修改操作会引起性能的降低

创建索引的一般建议

- WHERE子句中的条件含有AND操作符，最好创建一个多属性索引；
- 在连接操作的情况下，建议对每个连接列创建索引

Summary

- Stored Routine
- Views
- Index

多选题 1分

互动交流——不定项选择

MySQL中的存储程序包括(选择两项)

A Stored Routine

B Stored Procedure

C Stored Function

D Persistent Stored Modules

提交

多选题 1分

互动交流二——不定项选择

以下属于高级编程语言的数据库接口的是

A JDBC

B ODBC

C DB API

D Embeded SQL

提交

多选题 1分

互动交流三——不定项选择

以下说法正确的是

A 视图和数据库表一样，是实际存在的表，可以对数据提供安全保护

B 在创建视图时，必须包含SELECT语句

C 在创建视图时，可以不指定视图中的列名

D 在创建视图时，可以指定一部分视图中的列名

提交

主观题 10分

互动交流四

基于看到的数据库表，请创建视图使其内容为员工姓名以及对应工作地点。

- Department

deptno	deptname	location
d1	开发部	天津
d2	财务部	北京
d3	市场部	广东

- Employee

empno	empname	deptno
2581	徐唱	d2
9031	李静	d2
10102	王闻刚	d3
18316	冯新	d1
25348	张风	d3
28559	刘国风	d1
29346	赵东生	d2

提交

讲解

- Department

deptno	deptname	location
d1	开发部	天津
d2	财务部	北京
d3	市场部	广东

- Employee

empno	empname	deptno
2581	徐唱	d2
9031	李静	d2
10102	王闻刚	d3
18316	冯新	d1
25348	张风	d3
28559	刘国风	d1
29346	赵东生	d2

创建视图使其内容为员工姓名及对应工作地点。

```
CREATE VIEW em_location
 AS SELECT empname, location
FROM employee,department
 where employee.deptno=department.deptno
```

多选题 1分

互动交流五——不定项选择

对于刚才创建自Department和Employee表的视图em_location，以下说法正确的是

A 对Department表的更改一定会影响视图em_location的内容

B 对Employee表empname属性的更改一定会影响视图em_location的内容

C 视图em_location是可更新的视图

D 对视图em_location的更改会影响Department和Employee表的内容

提交

主观题 10分

互动交流六

对于关系workson(empno,projectno,job,enterdate),创建一个由主键属性构成的索引的SQL语句是

提交

讲解

对于关系workson(empno,projectno,job,enterdate),创建一个由主键属性构成的索引的SQL语句是

CREATE INDEX KeyIndex ON workson (empno,projectno);

SQL Authorization

Privileges

Grant and Revoke

Grant Diagrams

Privileges --- 1

- SQL identifies a more detailed set of privileges on objects (relations) than the typical file system.
- Nine privileges in all, some of which can be restricted to one column of one relation.

Privileges --- 2

- Some important privileges on a relation:
  1. SELECT = right to query the relation.
     - May apply to only one attribute.
  2. INSERT = right to insert tuples.
     - May apply to only one attribute.
  3. DELETE = right to delete tuples.
  4. UPDATE = right to update tuples.
     - May apply to only one attribute.

Granting Privileges

- You have all possible privileges on the objects, such as relations, that you create.
- You may grant privileges to other users (authorization ID’s), including PUBLIC.
- You may also grant privileges WITH GRANT OPTION, which lets the grantee also grant this privilege.

The GRANT Statement

- To grant privileges, say:

  GRANT <list of privileges>

  ON <relation or other object>

  TO <list of authorization ID’s>;

- If you want the recipient(s) to be able to pass the privilege(s) to others add:

  WITH GRANT OPTION

Example: GRANT

- Suppose you are the owner of Sells. You may say:

```
GRANT SELECT, UPDATE(price)
        ON Sells
        TO sally;
```

- Now Sally has the right to issue any query on Sells and can update the price component only.

Example: Grant Option

- Suppose we also grant:

```
GRANT UPDATE ON Sells TO sally
WITH GRANT OPTION;
```

- Now, Sally can not only update any attribute of Sells, but can grant to others the privilege UPDATE ON Sells.
  - Also, she can grant more specific privileges like `UPDATE(price) ON Sells`.

Revoking Privileges

REVOKE <list of privileges>
ON <relation or other object>
FROM <list of authorization ID's>;

- Your grant of these privileges can no longer be used by these users to justify their use of the privilege.
  - But they may still have the privilege because they obtained it independently from elsewhere.

REVOKE Options

- We must append to the REVOKE statement either:
  1. CASCADE. Now, any grants made by a revoke are also not in force, no matter how far the privilege was passed.
  2. RESTRICT. If the privilege has been passed to others, the REVOKE fails as a warning that something else must be done to "chase the privilege down."

Grant Diagrams

- Nodes = user/privilege/option/isOwner?
  - UPDATE ON R, UPDATE(a) on R, and UPDATE(b) ON R live in different nodes.
  - SELECT ON R and SELECT ON R WITH GRANT OPTION live in different nodes.
- Edge *X* ->*Y* means that node *X* was used to grant *Y*.

Notation for Nodes

- Use *AP* for the node representing authorization ID *A* having privilege *P*.
  - *P* * represents privilege *P* with grant option.
  - *P* ** represents the source of the privilege *P*. That is, *AP* ** means *A* is the owner of the object on which *P* is a privilege.
    - Note ** implies grant option.

Manipulating Edges --- 1

- When *A* grants *P* to *B*, we draw an edge from *AP* * or *AP* ** to *BP*.
  - Or to *BP* * if the grant is with grant option.
- If *A* grants a subprivilege *Q* of *P* (say UPDATE(a) on R when *P* is UPDATE ON R) then the edge goes to *BQ* or *BQ* *, instead.

Manipulating Edges --- 2

- Fundamental rule: user *C* has privilege *Q* as long as there is a path from *XQ* ** (the origin of privilege *Q*) to *CQ*, *CQ* *, or CQ**.
  - Remember that XQ** could *be* CQ**.

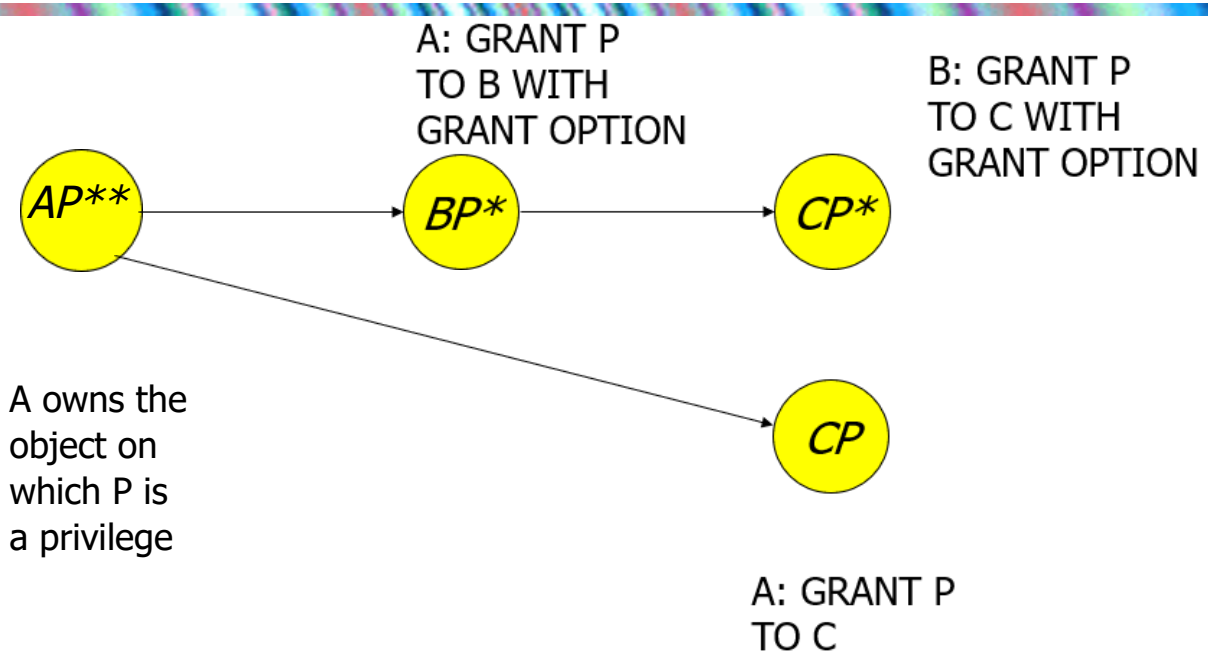
Manipulating Edges --- 3

- If *A* revokes *P* from *B* with the CASCADE option, delete the edge from *AP* to *BP*.
- If *A* uses RESTRICT, and there is an edge from *BP* to anywhere, then reject the revocation and make no change to the graph.

Manipulating Edges --- 4

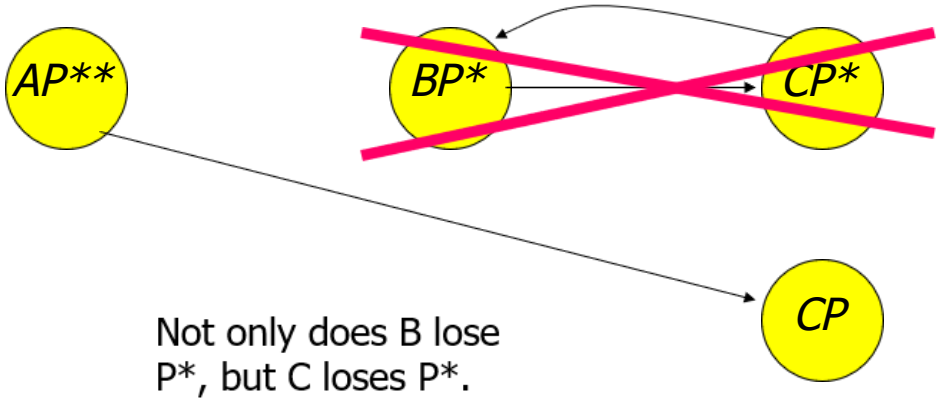
- Having revised the edges, we must check that each node has a path from some ** node, representing ownership.
- Any node with no such path represents a revoked privilege and is deleted from the diagram.

Example: Grant Diagram



Example: Grant Diagram

A executes
REVOKE P FROM B CASCADE;



Even had C passed P to B, both nodes are still cut off.

Not only does B lose P*, but C loses P*. Delete BP* and CP*.

However, C still has P without grant option because of the direct grant.

Summary

- Privileges
- Grant
- Revoke
- Grant Diagram

多选题 1分

互动交流——不定项选择

SQL中的权限包括

A SELECT FROM

B UPDATE

C DELETE FROM

D INSERT INTO

提交

多选题 1分

互动交流二——不定项选择

在SQL语言中，使用权的授予使用命令

A GRANT

B REVOKE

C GIVE

提交

多选题 1分

互动交流三——不定项选择

假如关系Sells(bar,beer,price)的拥有者执行如下SQL语句：
GRANT SELECT ON Sells TO Sally WITH GRANT OPTION;
GRANT UPDATE ON Sells TO PUBLIC; 则Sally

A 可以对sells表进行查询

B 可以执行GRANT SELECT(price) ON Sells TO PUBLIC;

C 可以对sells表进行更新

D 可以执行GRANT UPDATE(price) ON Sells TO PUBLIC;

提交

多选题 1分

互动交流四——不定项选择

假如关系Sells(bar,beer,price)的拥有者Bella执行过如下SQL语句：
GRANT SELECT ON Sells TO Sally WITH GRANT OPTION;

若Bella想要执行SQL语句 REVOKE SELECT ON Sells FROM Sally; 则

A 如果Sally已经将SELECT权限授予他人，则权限全部被收回

B 如果Sally已经将SELECT权限授予他人，则此SQL语句不被执行

C 如果Sally没有将SELECT权限授予他人，则Sally的SELECT权限被收回

D 这是错误的SQL语句

提交

单选题 1分

互动交流五

Initially, user A is the owner of relation R, and no other user holds privileges on R. The following are executed:

by A: GRANT UPDATE ON R TO B

by A: GRANT UPDATE(a) ON R TO C WITH GRANT OPTION

by C: GRANT UPDATE(a) ON R TO B WITH GRANT OPTION

by A: REVOKE UPDATE(a) ON R FROM C CASCADE

Which of the following best describes the status of B's privileges on R?

A B can update any attribute of R except a but cannot grant that privilege.

B B has no privileges on R and cannot grant any.

C B can update any attribute of R except a, but can grant others the privilege to update R:a .

D B can perform any update on R but cannot grant that privilege.

提交

讲解

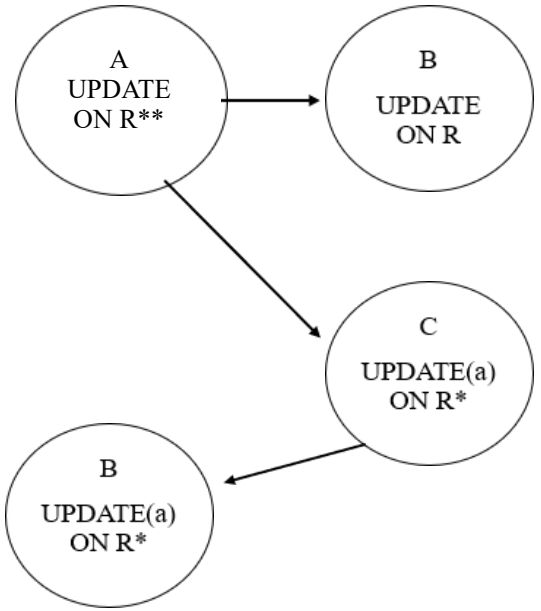
Initially, user A is the owner of relation R, and no other user holds privileges on R. The following are executed:

by A: GRANT UPDATE ON R TO B

by A: GRANT UPDATE(a) ON R TO C WITH GRANT OPTION

by C: GRANT UPDATE(a) ON R TO B WITH GRANT OPTION

by A: REVOKE UPDATE(a) ON R FROM C CASCADE



单选题 1分

互动交流六

Consider a database with relation R and users Alice, Bob, Carol, and Dave. Alice owns relation R. The following sequence of operations takes place:

Alice: GRANT SELECT ON R TO Bob WITH GRANT OPTION

Alice: GRANT SELECT ON R TO Carol WITH GRANT OPTION

Carol: GRANT SELECT ON R TO Bob WITH GRANT OPTION

Bob: GRANT SELECT ON R TO Dave WITH GRANT OPTION

Carol: GRANT SELECT ON R TO Dave

Dave: GRANT SELECT ON R TO Carol WITH GRANT OPTION

Alice: REVOKE SELECT ON R FROM Bob CASCADE

After these statements are executed, which of the following statements is true?

A Dave has the SELECT ON R privilege, but without the grant option.

B Dave has the SELECT ON R privilege with the grant option.

C Dave does not have the SELECT ON R privilege.

D Dave has the grant option for the SELECT ON R privilege, but does not have the privilege itself.

提交

讲解

Consider a database with relation R and users Alice, Bob, Carol, and Dave. Alice owns relation R. The following sequence of operations takes place:

Alice: GRANT SELECT ON R TO Bob WITH GRANT OPTION

Alice: GRANT SELECT ON R TO Carol WITH GRANT OPTION

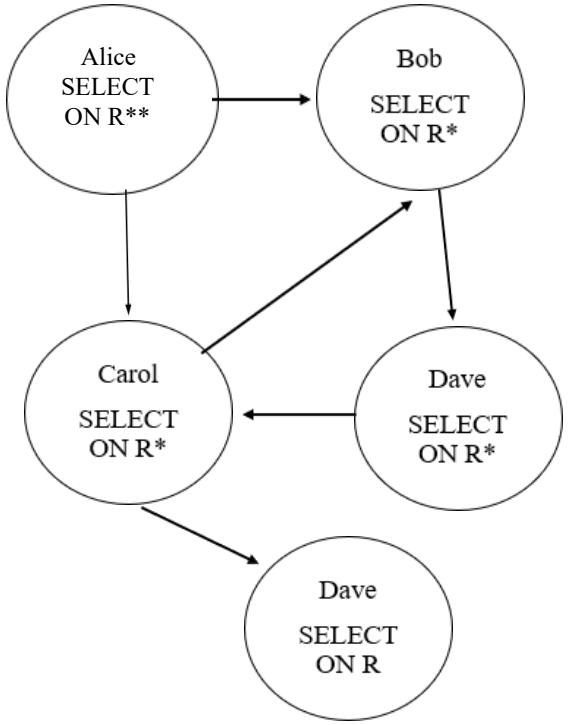
Carol: GRANT SELECT ON R TO Bob WITH GRANT OPTION

Bob: GRANT SELECT ON R TO Dave WITH GRANT OPTION

Carol: GRANT SELECT ON R TO Dave

Dave: GRANT SELECT ON R TO Carol WITH GRANT OPTION

Alice: REVOKE SELECT ON R FROM Bob CASCADE



主观题 10分

互动交流七

假设A是权限p所涉关系的属主，则

步骤	经由	操作
1	A	GRANT p TO B WITH GRANT OPTION
2	A	GRANT p TO C
3	B	GRANT p TO D WITH GRANT OPTION
4	D	GRANT p TO B,C,E WITH GRANT OPTION
5	B	REVOKE p FROM D CASCADE
6	A	REVOKE p FROM C CASCADE

以上步骤后的授权图是什么样子?

提交

讲解

- 1 A GRANT p TO B WITH GRANT OPTION
- 2 A GRANT p TO C
- 3 B GRANT p TO D WITH GRANT OPTION
- 4 D GRANT p TO B, C, E WITH GRANT OPTION
- 5 B REVOKE p FROM D CASCADE
- 6 A REVOKE p FROM C CASCADE