1. Chapter 1:
2. Q1
3. 数据: 描述事物的符号记录; 数据的种类有数字文字图形图像声音正文等;
4. 数据库: 长期储存在计算机内, 有组织的, 可共享的大量数据的集合; 数据库中的数据按一定的数据模型组织描述和储存，具有较小的冗余度，较高的数据独立性和易扩展性，并可为各种用户共享;
5. 数据库管理系统: 位于用户和操作系统之间的一层数据管理软件, 用于科学地组织和存储数据, 高效地获取和维护数据;
6. 数据库系统: 由数据库，数据库管理系统，应用程序和数据库管理员组成的存储，管理，处理和维护数据的系统;
7. Q2
8. 大大提高应用开发的速率, 方便用户的使用, 减轻数据库系统管理人员维护的负担;
9. Q3
10. 数据结构化; 数据的共享性高，冗余度低且易扩充; 数据独立性高; 数据由数据库管理系统统一管理和控制;
11. Q4
12. 数据的备份, 软件或应用程序使用过程中的临时数据存储一般使用文件;
13. 一个工厂的信息管理系统;
14. Q5
15. 数据结构化; 数据的共享性高，冗余度低且易扩充; 数据独立性高; 数据由数据库管理系统统一管理和控制;
16. Q6
17. 数据定义功能; 数据组织，存储和管理; 数据操纵功能; 数据库的事务管理和运行管理; 数据库的建立和维护功能; 其他功能;
18. Q7
19. 概念模型用于信息时解的建模，是现实世界到信息时解的第一层抽象，是数据库设计人员进行数据库设计的有力工具，也是数据库设计人员和用户之间进行交流的语言;
20. Q8
21. 实体：客观存在并相互区别的事物;
22. 实体型：具有相同属性的实体必然具有共同的特征和性质; 用实体名及其属性名集合来抽象和刻画同类实体，称为实体型;
23. 实体集：同一类型实体的集合;
24. 联系：在现实世界中，事物内部以及事物之间是有联系的，这些联系在信息世界中反映为实体内部的联系和实体之间的联系;
25. Q9
26. 数据模型的概念：数据模型是严格定义的一族概念的集合;
27. 数据模型的作用：这些概念精确地描述了系统的静态特性，动态特性和完整性约束条件;
28. 三个要素：数据结构，数据操作，数据的完整性约束条件;
29. Q10
30. 层次模型用树形结构来表示各类实体以及实体间的联系; 教员学生层次数据库模型; 行政机构层次数据库模型; 行政区域层次数据库模型;
31. Q11
32. 允许一个以上结点无双亲, 一个结点可以有多于一个的双亲; 人种植砍伐树; 父母养育赡养子女; 学生课程选课;
33. Q12
34. 层次优点: 层次模型的数据结构比较简单清晰; 层次数据库的查询效率高; 层次数据模型提供了良好的完整性支持;
35. 层次缺点：现实世界中很多联系是非层次性的; 如果一个结点具有多个双亲结点，用层次模型表示这类联系就很笨拙; 查询子女结点必须通过双亲结点; 由于结构严密，层次命令趋于程序化;
36. 网状优点：能够更为直接地描述现实世界; 具有良好的性能，存取效率较高;
37. 网状缺点：结构比较复杂，不利于最终用户掌握; 网状模型的数据定义语言DDL，数据操作语言DML复杂，并且要嵌入某一种高级语言中，用户不容易掌握，不容易使用; 由于记录之间的联系是通过存取路径实现的，应用程序在访问数据时必须选择适当的存取路径，因此用户必须了解系统结构的细节，加重了编写应用程序的负担;
38. Q13
39. 关系数据模型由关系数据结构，关系操作集合和关系完整性约束三部分组成。在用户观点下，关系数据模型中数据的逻辑结构是一张二维表，它由行和列组成；
40. 关系：一个关系对应通常说的一张表;
41. 属性：表中的一列即为一个属性，给每一个属性起一个名称即属性名;
42. 域：域是一族具有相同数据类型的值的集合;
43. 元组：表中的一行即为一个元组;
44. 码：表中的某个属性组，它可以唯一确定一个元组;
45. 分量：元组中的一个属性值;
46. 关系模式：对关系的描述;
47. Q14
48. 优点：关系模型与格式化模型不同，它是建立在严格的数学概念的基础上的; 关系模型的概念单一; 关系模型的存取路径对用户透明，从而具有更高的数据独立性，更好的安全保密性，也简化了程序员的工作和数据库开发建立的工作;
49. 缺点：查询效率不如格式化数据模型; 必须对用户的查询请求进行优化，增加了开发数据库管理系统的难度;
50. Q15
51. 模式：数据库中全体数据的逻辑结构和特征的描述，是所有用户的公共数据视图;
52. 外模式：数据库用户能够看见和使用的局部数据的逻辑结构和特征的描述，是数据库用户的数据是图，是与某一应用有关的数据的逻辑表示;
53. 内模式：数据物理结构和存储方式的描述，是数据在数据库内部的组织方式;
54. 它把数据的具体组织留给数据库管理系统管理，使用户能逻辑地，抽象地处理数据，而不必关系数据在计算机中的具体表示方式与存储方式;
55. Q16
56. 模式：数据库中全体数据的逻辑结构和特征的描述，是所有用户的公共数据视图;
57. 外模式：数据库用户能够看见和使用的局部数据的逻辑结构和特征的描述，是数据库用户的数据是图，是与某一应用有关的数据的逻辑表示;
58. 内模式：数据物理结构和存储方式的描述，是数据在数据库内部的组织方式;
59. 数据定义语言是用来定义数据库模式，外模式，内模式的语言;
60. 数据操纵语言是用来对数据库中的数据进行查询，插入，删除和修改的语言;
61. Q17
62. 当模式改变时，由数据库管理员对各个外模式/模式的映像作相应改变，可以使外模式保持不变; 应用程序是依据数据的外模式编写的，从而应用程序不必修改，保证了数据与程序的逻辑独立性，简称数据的逻辑独立性;
63. 当数据库的存储结构改变时，由数据库管理员对模式/内模式映像作相应改变，可以使模式保持不变，从而应用程序也不必改变；保证了数据与程序的物理独立性，简称数据物理独立性;
64. 数据和程序之间的独立性使得数据的定义和描述可以从应用程序中分离出去；另外，由于数据的存取由数据库管理系统管理，从而简化了应用程序的编制，大大减少了应用程序的维护和修改;
65. Q18
66. 硬件平台及数据库; 软件; 人员;
67. Q19
68. 数据库管理员：决定数据库中的信息内容和结构; 决定数据库的存储结构和存取策略; 监控数据库的使用和运行; 数据库的改进和重组，重构;
69. 系统分析员：负责应用系统的需求分析和规范说明，要和用户及数据库管理员相结合，确定系统的硬件资源配置，并参与数据库系统的概要设计;
70. 数据库设计人员：负责数据库中数据的确定及数据库各级模式的设计；数据库设计人员必须参加用户需求调查和系统分析，然后进行数据库设计;
71. 应用程序员：负责设计和编写应用系统的程序模块，并进行调试和安装;
72. Chapter 2
73. Q1
74. 关系数据结构; 关系操作集合; 关系完整性约束;
75. Q2
76. 关系数据语言可以分为三类：关系代数语言; 关系演算语言，元组关系演算语言和域关系演算语言; SQL，具有关系代数和关系演算双重特点的语言; 语言具有完备的表达能力，是非过程化的集合操作语言，功能强，能够嵌入高级语言中使用；
77. Q3
78. 域：一组具有相同数据类型的值的集合;
79. 笛卡儿积：域上的一种集合运算;
80. 关系：笛卡儿积的有限子集;
81. 元组：每一个元素叫做一个元组;
82. 属性：必须对每列起一个名字;
83. 主码：若一个关系有多个候选码，则选定其中一个为主码;
84. 候选码：若关系中的某一属性组的值能唯一标识一个元组，而且其子集不能，则称该属性组为候选码;
85. 外码：如果公共关键字在一个关系中是主关键字，那么这个公共关键字被称为另一个关系的外键;
86. 关系模式：关系的描述称为关系模式;
87. 关系数据库：关系数据库有型和值之分；关系数据库的型也称为关系数据库模式，是对关系数据库的描述，它包括若干域的定义以及在这些域上定义的若干关系模式；关系数据库的值是这些关系模式在某一时刻对应的关系的集合，通常称为关系数据库;
88. Q4
89. 关系数据库中，关系模式是型，关系是值; 关系模式是对关系的描述;
90. Q5
91. 关系模型的完整性规则是对关系的某种约束条件; 实体完整性，参照完整性，用户定义完整性; F的每个属性值是空值; 属性F本身不是主属性，则可以取空值，否则不能取空值;
92. Q6
93. ？？？？;;;;;;;;;;;;;;;;;;;;;;
94. Q7
95. Theta为=的连接运算称为等值连接；它是从关系R与S的广义笛卡儿积中选取AB属性值相等的那些元组；
96. 自然连接是一种特殊的等值连接；他要求两个关系中进行比较的分量必须是同名的属性组，并且在结果中把重复的属性列去掉；
97. Q8
98. 基本运算：并，差，笛卡儿积，投影，选择；
99. R交S = R - (R-S)
100. R连接S = 选R[A] = S[A](R x S)
101. R / S = 投影X(R) - 投影X(投影X(R) \* 投影Y(S) - R)；
102. Chapter 3
103. Q1
104. 综合统一；高度非过程化；面向集合的操作方式；以同一种语法结构提供多种使用方式；语言简洁，易学易用；
105. Q2
106. 若选择restrict，则该表的删除是有限制条件的。欲删除的基本表不能被其他表的约束所引用，不能有视图，不能有触发器，不能有存储过程或函数等。如果存在这些依赖该表的对象，则此表不能被删除。默认情况是restrict。
107. 若选择cascade，则该表的删除没有限制条件。在删除基本表的同时，相关的依赖对象，例如视图，都将一起被删除。
108. Q3
109. (1) Select \* from S where A = ‘10’
110. (2) select A, B from S
111. (3) select A, B, SC, SD, E, F from S, T
112. (4) select \* from S, T where SC = TC
113. (5) select \* from S, T where SA<TE
114. (6) select SC, SD, T.\* from S, T
115. Q4
116. (1) create table s(sno char(3), sname char(10), status char(2), city char(10))
117. Create table p(PNO char(3), pname char(10), color char(4), weight int);
118. Create table j(jno char(3), jname char(10), city char(10));
119. Create table spj(sno char(3), pno char(3), jno char(3), qty int);
120. (2) select sno from spj where jno=’j1’
121. (3) select sno from spj where jno=’j1’ and pno=’p1’
122. (4) select sno from P, SPJ where JNO=‘J1’ AND spj.pno=p.pno AND color=’红’
123. (5) select jno from j where not exists (select \* from spj where spj.jno=j.jno AND sno in (select sno from s where city=’天津’) and pno in (select pno from p where color=’红’))
124. (6) select jno from j where ？？？？？？？
125. Q5
126. (1) select sname, city from s;
127. (2) select pname, color, weight from p;
128. (3) select jno from spj where sno=’s1’;
129. (4) select p.pname, spj.qty from spj, p where JNO=’j2’ and spj.pno=p.pno;
130. (5) select distinct pno from spj where sno in (select sno from s where city=’上海’);
131. (6) select jname from j, s, spj where s.city=’上海’ and s.sno=spj.sno and j.jno=spj.jno;
132. (7) select jno from j where not exists (select \* from spj,s where spj.jno=j.jno and spj.sno=s.sno and s.city=’天津’);
133. (8) update p set color=’蓝’ where color=’红’;
134. (9) update spj set sno=’s3’ where sno=’s5’ and jno=’j4’ and pno=’p6’;
135. (10) delete from spj where sno=’s2’; delete from s where sno=’s2’;
136. (11) insert into spj(sno, jno, pno, qty) values(‘s2’, ‘j6’, ‘P4’, 200);
137. Q6
138. 基本表：是独立存在的表；在SQL中，一个关系对应于一个表；
139. 视图：是从一个或多个基本表所导出的表；视图本身并不独立存储在数据库中，是一个虚表，即数据库中只存放视图的定义而不存放其所对应的数据，这些数据仍然存放在到处的视图的基本表中；视图在概念上与基本表等同，用户可以像使用基本表那样使用视图，可以在视图上再定义视图；
140. Q7
141. 视图能够简化用户的操作；视图使用户能以多种角度看待同一数据；视图对重构数据库提供了一定程度的逻辑独立性；能够对机密数据提供安全保护；适当利用视图可以更清晰地表达查询；
142. Q8
143. 行列子集视图是可以更新的；除行列子集视图以外，有些视图理论上是可更新的，但它们的确切特征还是尚待研究的课题；还有些视图从理论上就是不可更新的，因为有些视图的更新不能唯一有意义地转换成相应基本表的更新；
144. Q9
145. Create view v\_spj as select sno, pno, qty from spj where jno=(select jno from j where jname=’三建’)；
146. (1) select pno, qty from v\_spj;
147. (2) Select pno, qty from v\_spj where sno=’s1’;
148. Chapter 4
149. Q1
150. 数据库的安全性是指保护数据库以防止不合法使用所造成的数据泄漏，更改和破坏；
151. Q2
152. 非授权用户对数据库的恶意存取和破坏；数据库中重要或敏感的数据被泄漏；安全环境的脆弱性；
153. Q3
154. EAL１，功能测试；EAL2，结构测试；EAL3，系统地测试和检查；EAL4，系统地设计，测试和检查；EAL5，半形式化设计和测试；EAL6，半形式化验证的设计和测试；EAL7，形式化验证呢光的设计和测试；
155. Q4
156. 用户身份鉴别，静态口令鉴别，动态口令鉴别，生物特征鉴别，智能卡鉴别；存取控制；自主存取控制方法；授权：授予与收回，GRANT，REVOKE；数据库角色；强制存取控制方法；
157. Q5
158. 自主存取控制方法：定义各个用户对不同数据对象的存取权限；当用户对数据库访问时首先检查用户的存取权限；防止不合法用户对数据库的存取；
159. 强制存取控制方法：每一个数据对象被强制地标以一定的保密等级，每一个用户也被授予某一个级别的许可证；系统规定只有具有某一许可证级别的才能存取某一个密级的数据对象；
160. Q6
161. (1) grant all privileges on student, class to u1 with grant option;
162. (2) grant select, update(addr) on table student to u2;
163. (3) grant select on class to public;
164. (4) grant select, update on student to R1;
165. (5) grant R1 to U1 with grant option;
166. Q7
167. (1) grant select on zhigong, bumen to WangMing;
168. (2) grant insert, delete on zhigong, bumen to LiYong;
169. (3) grant select on zhigong when user()=name to all;
170. (4) grant select, update(gongzi) on zhigong to LiuXing;
171. (5) grant alter table on zhigong, bumen to ZhangXin;
172. (6) grant all privileges on zhigong, bumen to ZhouPing with grant option;
173. (7) create view 部门工资 as select 部门.名称，Max(工资), Min(工资), Avg(工资) from 职工，部门，Max(工资), Min(工资)，AVG(工资) ??????
174. Q8
175. (1) revoke select on zhigong, bumen from WangMing;
176. (2) revoke insert, delete on zhigong, bumen from LiYong;
177. (3) revoke select on zhigong when user()=name from all;
178. (4) revoke select, update(gongzi) on zhigong from LiuXing;
179. (5) revoke alter table on zhigong, bumen from ZhangXin;
180. (6) revoke all privileges on zhigong, bumen from ZhouPing;
181. (7) revoke select on 部门工资 from YangLan; drop view 部门工资；
182. Q9
183. 主体是系统中的活动实体，既包括数据库管理系统所管理的实际用户，也包括代表用户的各进程；
184. 客体是系统中的被动实体，是受主体操纵的，包括文件，基本表，索引，视图等；
185. 对于主体和客体，数据库管理系统为它们每个实例指派一个敏感度标记；主体的敏感度标记称为许可证级别，客体的敏感度标记成为密级；
186. Q10
187. 仅当主体的许可证级别大于或等于客体的密级时，该主体才能读取相应的客体；仅当主体的许可级别小于客体的密级时，该主体才能写相应的客体；
188. Q11
189. 审计功能把用户对数据库的所有操作自动记录下来放入审计日志中；审计员可以利用审计日志监控数据库中的各种行为，重现导致数据库现有状况的一系列事件，找出非法存取数据的人，时间和内存等；
190. Chapter 5
191. Q1
192. 数据库的完整性是指数据的正确性和相容性；数据的正确性是指数据是符合现实世界语义，反映当前实际状况的；数据的相容性是指数据库统一对象在不同关系表中的数据是符合逻辑的；
193. Q2
194. 数据的完整性和安全性是两个既有联系又不尽相同的概念；数据的完整性是为了防止数据库中存在不符合意义的数据，也就是防止数据库中存在不正确的数据；数据的安全性是保护数据库防止恶意破坏和非法存取；因此，完整性检查和控制的防范对象是不合语义的，不正确的数据，防止它们进入数据库；安全性控制的防范对象是非法用户和非法操作，防止他们对数据库数据的非法存取；
195. Q3
196. 完整性约束条件也称为完整性规则，是数据库中的数据必须满足的语义约束条件；它表达了各定的数据模型中数据及其联系所具有的制约和依存规则，用以限定符合数据模型的数据库状态以及状态的变化，以保证数据的正确，有效和相容；SQL标准使用了一系列概念来描述完整性，包括关系模型的实体完整性，参照完整性和用户定义完整性；这些完整性一般由SQL数据定义语言语句来实现，它们作为数据库模式的一部分存入数据字典中；
197. Q4
198. 提供定义完整性约束条件的机制；提供完整性检查的方法；进行违约处理；
199. Q5
200. 外码是否可以接受空值；
201. 被参照关系的元组时的考虑，这是系统可能采取的做法有三种，级联删除cascades， 受限删除restricted, 置空值删除nullifies;
202. 在参照关系中插入元组时的问题，这时系统可能采取的做法有，受限插入，递归插入；
203. 修改关系中主码的问题；一般是不能用update语句修改关系主码的；如果需要修改主码值，只能先删除该元组，然后再把具有新主码值的元组插入到关系中；如果允许修改主码，首先要保证主码的唯一性和非空性，否则拒绝修改；然后要区分是参照关系还是被参照关系；
204. Q6
205. Create table 职工 (职工号 char(3) not null, 姓名 char(3) not null, 年龄 int not null check(年龄<=60), 职务 char(3) not null, 工资 int not null, foreign key (部门号) references 部门(部门号), primary key(职工号));
206. Create table 部门(部门号 char(9) not null, 名称 char(9) not null, 经理名 char(9) not null, 电话 char(9) not null, primary key(部门号));
207. Q7
208. 对于违反实体完整性和用户定义的完整性的操作一般都采用拒绝执行的方式进行处理；而对于违反参照完整性的操作，并不都是简单拒绝执行，有时要根据语义执行一些附加的操作，以保证数据库的正确性；
209. Q8
210. Create table Male(age int);
211. Create table Female(age int);
212. Create assertion 人数限制 check (50>= select count (\*) from (select \* from Male Union Select \* from female) as guest);
213. Chapter 6
214. Q1
215. 函数依赖：设R(U)是属性集U上的关系模式，XY是U的子集；若对于R(U)的任意一个可能的关系r, r中不可能存在两个元组在X上的属性值相等，而在Y上的属性值不等，则称X函数确定Y或Y函数依赖于X，则称X函数确定Y或Y函数依赖于X，记作X->Y;
216. 部分函数依赖：若X->Y, 但Y不完全依赖于X，则称Y对X部分函数依赖；
217. 完全函数依赖：并且对于X的任何一个真子集X‘，则称Y对X完全函数依赖；
218. 传递依赖：如果X->Y, Y推不出X，Y->Z, 则称Z对X传递函数依赖；
219. 候选码：设K为R(U, F)中的属性或属性组合，若K->(F)U, 则K为R的候选码；
220. 超码：如果U函数依赖于K，即K->U，则K称为超码；
221. 主码：若候选码多于一个，则选定一个为主码；
222. 外码：关系模式R中属性或属性组X并非R的码，但X是另一个关系模式的码，则称X是R的外部码，也称外码；
223. 全码：最极端的情况，整个属性组都是码，称为全码；
224. 1NF：作为一个二维表，关系要符合一个最基本的条件，每一个分量必须是不可分的数据项，满足了这个条件的关系模式就属于第一范式1NF；
225. 2NF：每一个非主属性完全函数依赖于任何一个候选码；
226. 3NF：不存在码X，属性组Y及非主属性Z，使得X->Y, Y->Z成立，Y推不出X；
227. BCNF：若X->Y且Y不是X子集时X必含有码；
228. 多值依赖：设R(U)是属性集U上的一个关系模式；XYZ是U的子集，并且Z=U-X-Y；关系模式R(U)中多值依赖X->->Y成立，当且仅当R(U)中的任一关系r，给定的一对(x, z)值，有一组Y的值，这组值仅仅决定于x值而与z值无关；
229. 4NF：不允许有非平凡且非函数依赖的多值依赖；
230. Q2
231. ？？？？？？？？？？？？？？？？？？？？？？？？
232. ？？？？？？？？？？？？？？？？？？？？？？？？？？
233. ？？？？？？？？？？？？？？？？？？？？？？？？？？？？
234. ？？？？？？？？？？？？？？？？？？？？？？？？？？？？？？
235. Q3
236. Q4
237. Q5
238. Q6
239. Q7
240. Q8
241. Chapter 7
242. Q1
243. 需求分析阶段；概念结构设计阶段；逻辑结构设计阶段；物理结构设计阶段；数据库实施阶段；数据库运行和维护阶段；
244. Q2
245. 在需求分析阶段综合各个用户的应用需求；在概念结构设计阶段形成独立于机器特点，独立于各个关系数据库管理系统产品的概念模式，在本篇中就是ER图；在逻辑结构设计阶段将ER图转换成具体的数据库产品支持的数据模型，如关系模型，形成数据库逻辑模式，然后根据用户处理的要求，安全性考虑，在基本表的基础上再建立必要的视图，形成数据的外模式；在物理结构设计阶段，根据关系数据库管理系统的特点和处理的需要进行物理存储安排，建立索引，形成数据库内模式；
246. Q3
247. 需求分析的任务是通过详细调查现实世界要处理的对象，充分了解原系统的工作概况，明确用户的各种需求，然后在此基础上确定新系统的功能；新系统必须充分考虑今后可能的扩充和改变，不能仅仅按当前应用需求来设计数据库；
248. 调查的重点是数据和处理，通过调查，收集和分析，获得用户对数据库的如下要求；信息要求，指用户需要从数据库中获得信息的内容与性质，由信息要求可以导出数据要求，即在数据库中需要存储哪些数据；处理要求，指用户要完成的数据处理功能，对处理性能的要求；安全性与完整性要求；
249. Q4
250. 数据字典是进行详细的数据收集和数据分析所获得的主要成果；它是关于数据库中数据的描述，即元数据，而不是数据本身；数据字典是在需求分析阶段建立，在数据库设计过程中不断修改，充实，完善的；它在数据库设计中占有很重要的地位；
251. 数据字典通常包括数据项，数据结构，数据流，数据存储和处理过程几部分；其中数据项是数据的最小组成单位，若干个数据项可以组成一个数据结构；数据字典通过对数据项和数据结构的定义来描述数据流，数据存储的逻辑内容；
252. Q5
253. 在需求分析阶段所得到的应用需求应该首先抽象为信息世界的结构，然后才能更好更准确地用某一数据库管理系统实现这些需求；概念模型的主要特点是：
254. 能真实充分地反应现实世界，包括事物和事物之间的联系，能满足用户对数据的处理要求，是现实世界的一个真实模型；
255. 易于理解，可以用它和不熟悉计算机的用户交换意见；用户的积极参与是数据库设计成功的关键；
256. 易于更改，当应用环境和应用要求改变时容易对概念模型修改和扩充；
257. 易于向关系网状层次等各种数据模型转换；
258. 设计策略：自顶向下，即首先定义全局概念结构的框架，然后逐步细化；自底向上，即首先定义各局部应用的概念结构，然后将它们集成起来，得到全局概念结构；逐步扩张，首先定义最重要的核心概念结构，然后向外扩充，直至得到总体概念结构；混合策略，用自顶向下策略设计一个全局概念结构的框架，以它为骨架集成由自底向上策略中设计的各局部概念结构；
259. Q6
260. 实体：客观存在并可相互区别的事物称为实体；
261. 实体型：具有相同属性的实体必然具有共同的特征和性质；
262. 实体集：同一类型的集合称为实体集；
263. 属性：实体所具有的某一特性称为属性；
264. 码：唯一标识实体的属性集称为码；
265. ER图：ER图提供了表示实体型，属性和联系的方法；实体型用矩形表示，矩形框内写明实体名；属性用椭圆形表示，并用无向边将其与相应的实体型连接起来；联系用菱形表示；
266. Q7 ？？？？？？
267. Q8 ？？？？？？
268. Q9
269. 逻辑结构设计的任务就是把概念结构设计阶段设计好的基本ER图转换为与选用数据库管理系统产品所支持的数据模型相符合的逻辑结构；
270. ER图向关系模型的转换；数据模型的优化；设计用户字模式；
271. Q10 ？？？？？？？
272. Q11 ？？？？？？？
273. Q12
274. 规范化理论为数据库设计人员判断关系模式的优劣提供了理论标准，可用以指导关系数据模型的优化，用来预测模式可能出现的问题，为设计人员提供了自动产生各种模式的算法工具，使数据库设计工作有了严格的理论基础；
275. Q13
276. 为一个给定的逻辑数据模型选取一个最适合应用要求的物理结构的过程，就是数据库的物理设计；数据库的物理设计通常分两步：
277. 确定数据库的物理结构，在关系数据库中主要指存取方法和存储结构；
278. 对物理结构进行评价，评价的重点是时间和空间效率；
279. Q14
280. 一般数据库系统中数据量都很大，而且数据来源于部门中的各个不同的单位，数据的组织方式，结构和格式都与新设计的数据库系统有相当的差距；组织数据载入就要将各类源数据从各个局部应用中抽取出来，输入计算机，再分类转换，最后综合成符合新设计的数据库结构的形式，输入数据库；因此这样的数据转换，组织入库的工作是相当费力，费时的；
281. 为提高数据输入工作的效率和质量，应该针对具体的应用环境设计一个数据录入子系统，由计算机来完成数据入库的任务；在源数据入库之前要采用多种方法对其进行检验，以防止不正确的数据入库，这部分的工作在整个数据输入子系统中是非常重要的；
282. Q15
283. 数据库运行一段时间后，由于记录不断增删改，将会使数据库的物理存储情况变坏，降低数据的存取效率，使数据库性能下降，这时数据库管理员就要对数据库进行重组织或部分重组织；关系数据库管理系统一般都提供数据重组织用的实用程序；在重组织过程中，按原设计要求重新安排存储位置，回收垃圾，减少指针链等，提高系统性能；
284. 数据库的重组织并不修改原设计的逻辑和物理结构，而数据库的重构造则不同，它是指部分修改数据库的模式和内模式；
285. 由于数据库应用环境发生变化，增加了新的应用或新的实体，取消了某些应用，有的实体与实体间的联系也发生了变化等；使原有的数据库设计不能满足新的需求，需要调整数据库的模式和内模式；
286. Chapter 8
287. ??????????????????????????????????????????????????????
288. Chapter 9
289. Q1
290. 关系系统的查询优化既是关系数据库管理系统实现的关键技术，又是关系系统的优点所在；它减轻了用户选择存取路径的负担；用户只要提出“干什么”，而不必指出“怎么干”；查询优化的优点不仅在于用户不必考虑如何最好地表达查询以获得较高的效率，而且在于系统可以比用户程序的优化做得更好；
291. 优化器可以从数据字典中获取许多统计信息，优化器可以根据这些信息做出正确的估算，选择高效地执行计划；
292. 如果数据库中的物理统计信息改变了，系统可以自动对查询进行重新优化以选择相适应的执行计划；
293. 优化器可以考虑数百中不同的执行计划；
294. 优化器包括了很多复杂的优化技术；
295. Q2
296. ？？？？？？？？？？？？？？？？？？？？
297. Chapter 10
298. Q1
299. 所谓事务是用户定义的一个数据库操作序列，这些操作要么全做，要么全不做，是一个不可分割的工作单位；
300. 原子性：要么都做，要么都不做；
301. 一致性：事务执行的结果必须是使数据库从一个一致性状态变到另一个一致性状态；
302. 隔离性：一个事务的执行不能被其他事务干扰；
303. 持续性：一个事务一旦提交，它对数据库中数据的改变就应该是永久性的；接下来的其他操作或故障不应该对其执行结果有任何影响；
304. 一致性；
305. Q2
306. 事务执行的结果必须是使数据库从一个一致性状态变到另一个一致性状态；如果数据库系统运行中发生故障，有些事务尚未完成就被迫中断，这些未完成事务对数据库所做的修改有一部分已写入物理数据库，这时数据库就处于一种不正确的状态，或者说不一致性状态；
307. Q3
308. 把对数据的修改写到数据库中和把表示这个修改的日志记录写到日志文件中是两个不同的操作；有可能在这两个操作之间发生故障，即这两个写操作只完成了一个；如果先写了数据库修改，而在运行记录中没有登记这个修改，则以后就就无法恢复这个修改了；如果先写日志，但没有修改数据库，按日志文件恢复时只不过是多执行一次不必要的UNDO操作，并不影响到数据库的正确性；所以为了安全，一定要先写日志文件，即首先把日志记录写到日志文件中，然后写数据库的修改；这就是先写日志文件的原则；
309. Q4
310. (1) T1重做，T3重做，T4回滚；
311. (2) T1重做，T3回滚；
312. (3) T1重做，T2回滚，T3回滚；
313. (4) T1重做，T2回滚；
314. Q5
315. (1) A=8, B=7, C=11;
316. (2) A=10, B=0, C=11;
317. (3) A=10, B=0, C=11;
318. (4) A=10, B=0, C=11;
319. (5) A=10, B=0, C=11;
320. (6) A=0, B=0, C=0;
321. Q6
322. 事务故障的恢复：事务故障是指事务在运行至正常终止点前被终止，这时恢复子系统应利用日志文件撤销此事务已对数据库进行的修改；事务故障的恢复是由系统自动完成的，对用户是透明的；步骤是：
323. 反向扫描日志文件，查找该事务的更新操作；对该事务的更新操作进行逆操作；继续反向扫描日志文件；如此处理下去，直到读到此事务的逆操作；
324. 系统故障的恢复：正向扫描日志文件（即从头扫描日志文件），找出在故障发生前已经提交的事务；同时找出故障发生时尚未完成的事物，将其事务标识记入撤销队列；对撤销队列中的各个事务进行撤销处理；对重做队列中的各个事务进行重做处理；
325. 介质故障的恢复：发生介质故障后，磁盘上的物理数据和日志文件被破坏，这是最严重的一种故障，恢复方法是重装数据库，然后重做已完成的事务；装入最新的数据库后备副本，使数据库恢复到最近一次转储时的一致性状态；装入相应的日志文件副本，重做已完成的事务；即首先扫描日志文件，找出故障发生时已提交的事务的标识，将其记入重做队列；然后正向扫描日志文件，找出故障发生时已提交的事务的标识，将其记入重做队列；然后正向扫描日志文件，对重做队列中的所有事务进行重做处理；
326. Q7
327. 利用日志技术进行数据库恢复时，恢复子系统必须搜索日志，确定哪些事务需要重做，哪些事务需要撤销；一般来说，需要检查所有日志记录；这样做有两个问题，一是搜索整个日志将耗费大量的时间，二是很多需要重做处理的事务实际上已经将它们的更新操作结果写到了数据库中，然而恢复子系统又重新执行了这些操作，浪费了大量时间；为了解决这些问题，又发展了具有检查点的恢复技术；这种技术在日志文件中增加一类新的记录--检查点记录，增加一个重新开始文件，并让恢复子系统在登录日志文件动态地维护日志；
328. 检查点记录的内容包括：建立检查点时刻所有正在执行的事务清单；这些事务最近一个日志记录的地址；
329. Q8
330. 使用检查点可以改善恢复效率；当事务T在一个检查点之前提交，T对数据库所做的修改一定都写入数据库，写入时间是在这个检查点建立之前或者这个检查点建立之时；这样，在进行恢复处理时，没有必要对事务T执行重做操作；
331. Q9
332. 系统使用检查点方法进行恢复的步骤是：
333. 从重新开始文件中找到最后一个检查点记录在日志文件中的地址，由该地址在日志文件中找到最后一个检查点记录；由该检查点记录得到检查点建立时刻所有正在执行的事务清单ACTIVE-LIST; 这里建立两个事务队列，UNDO-LIST，需要执行UNDO操作的事务集合；REDO-LIST，需要执行REDO操作的事务集合；把ACTIVE-LIST暂时放入UNDO-LIST队列，REDO队列暂为空；从检查点开始正向扫描日志文件，如有新开始的事务Ti，把Ti暂时放入UNDO-LIST队列；如有提交的事务Tj，把Tj从UNDO-LIST队列移到REDO-LIST队列，直到日志文件结束；对UNDO-LIST中的每个事务执行UNDO操作，对REDO-LIST中的每个事务执行REDO操作；对UNDO-LIST中的每个事务执行UNDO操作，对REDO-LIST中的每个事务执行REDO操作；
334. Q10
335. 随着技术的发展，磁盘容量越来越大，价格越来越便宜；为避免磁盘介质出现故障影响数据库的可用性，许多数据库管理系统提供了数据库镜像mirror功能用于数据库恢复；即根据数据库管理员的要求，自动把整个数据库或其中的关键数据复制到另一个磁盘上，每当主数据库更新时，数据库管理系统自动把更新后的数据复制过去，由数据库管理系统自动保证镜像数据与主数据库的一致性；这样，一旦出现介质故障，可由镜像磁盘继续提供使用，同时数据库管理系统自动利用镜像磁盘数据进行数据库的恢复，不需要关闭系统和重装数据库副本；在没有出现故障时，数据库镜像还可以用于并发操作，即当一个用户对数据家排他锁修改数据时，其他用户可以读镜像数据库上的数据，而不必等待该用户释放锁；
336. Chapter 11
337. Q1
338. 数据库的一个明显的特点是多个用户共享数据库资源，尤其是多用户可以同时存取相同的数据；在这样的系统中，在同一时刻兵法云性的事务可达数百个；若对事务的并发操作不加以控制，就会造成数据存取的错误，破坏了数据的一致性和完整性；并发控制可以防止数据不一致性出现；
339. Q2
340. 并发操作带来的数据不一致性包括三类：丢失修改，不可重复读和读脏数据；
341. 丢失修改：两个事务T1和T2读入同一数据并修改，T2提交的结果破坏了T1提交的结果，导致T1的修改被丢失；
342. 不可重复读：事务T1读取数据后，事务T2执行更新操作，使T1无法再现前一次读取结果；
343. 读脏数据：事务T1修改某一数据，并将其写回磁盘，事务T2读取同一数据后，T1由于某种原因被撤销，这时T1已修改过的数据恢复原值，T2读到的数据就与数据库中的数据不一致，则T2读到的数据就为脏数据，即不正确的数据；避免不一致性的方法和技术就是并发控制；最常用的技术是封锁技术，也可以用其他技术，例如在分布式数据库系统中可以采用时间戳方法来进行并发控制；
344. Q3
345. 封锁就是事务T在对某个数据对象例如表记录等进行操作之前，先向系统发出请求，对其加锁。加锁后事务T就对该数据对象有了一定的控制，在事务T释放它的锁之前，其他的事务不能更新此数据对象；封锁是实现并发控制的一个非常重要的技术；
346. 基本的封锁类型有两种，排他锁和共享锁；排他锁又称为写锁；排他锁又称为写锁；若事务T对数据对象A加上X锁，则只允许T读取和修改A，其他任何事务都不能再对A加任何类型的锁，直到T释放A上的锁；这就保证了其他事务在T释放A上的锁之前不能再读取和修改A；
347. 共享锁又称为读锁；若事务T对数据A加上S锁，则事务T可以读A但不能修改A，其他事务只能再对A加S锁，而不能加X锁，直到T释放A上的S锁；这就保证了其他事务可以读A，但在T释放A上的S锁之前不能对A做任何修改；
348. Q4
349. 数据库管理系统在对数据进行读写操作之前首先对该数据执行封锁操作，例如下图中事务T1在对A进行修改之前先对A执行XLock(A), 即对A加X锁；这样，当T2请求对A加X锁时就被拒绝，T2只能等待T1释放A上的锁后才能获得对A的X锁，这时它读到的A是T1更新后的值，再按此新的A值进行运算；这样就不会丢失T1的更新；
350. Q5
351. 如果事务T1封锁了数据R，事务T2又请求封锁R，于是T2等待；T3也请求封锁R，当T1释放了R上的封锁之后系统首先批准了T3的请求，T2仍然等待......T2有可能永远等待，这就是活锁的情形；
352. 避免活锁的简单方法是采用先来先服务的策略；当多个事务请求封锁同一数据对象时，封锁子系统按请求封锁的先后次序对事务排队，数据对象上的锁一旦释放就批准申请队列中第一个事务获得锁；
353. Q6
354. 如果事务T1封锁了数据R1，T2封锁了数据R2，然后T1又请求封锁R2，因T2已经封锁了R2，于是T1等待T2释放R2的锁；接着T2又申请封锁R1，因T1已封锁了R1，T2也只能等待T1释放R1上的锁；这样就出现了T1在等待T2，而T2又在等待T1的局面，T1和T2两个事务永远不能结束，形成死锁；
355. 在数据库中，产生死锁的原因是两个或多个事务都已封锁了一些数据对象，然后又都请求已被其他事务封锁的数据加锁，从而出现死等待；
356. 防止死锁的发生其实就是要破坏产生死锁的条件，预防死锁通常有两种方法： (1) 一次封锁法，要求每个事务必须一次将所有要使用的数据全部加锁，否则就不能继续执行；(2) 顺序封锁法，预先对数据对象规定一个封锁顺序，所有事务都按这个顺序实行封锁；
357. Q7
358. 超时法：如果一个事务的等待时间超过了规定的时限，就认为发生了死锁；超时法实现简单，但有可能误判死锁，事务因其他原因长时间等待超过时限时，系统会误认为发生了死锁；若时限设置太长，又不能及时发现死锁发生；
359. 通常采用的方法是选择一个处理死锁代价最小的事务，将其撤销，释放此事务持有的所有锁，使其他事务得以继续运行下去；当然，对撤销的事务所执行的数据修改操作必须加以恢复；
360. Q8
361. 可串行化的调度是正确的调度；可串行化调度的定义，多个事务的并发执行是正确的，当且仅当其结果与按某一次序串行执行它们时的结果相同，称这种调度策略为可串行化的调度；
362. Q9
363. (1) 2, 4, 8, 16;
364. (2) slock A; Y=A=0; Unlock A; Xlock A; A = Y+2; 写回A; Unlock A;
365. -