From Postquel To Sql

Andrew Yu and Jolly Chen
Computer Science Division
University of California, Berkeley
May 1, 1995

POSTQUEL	SQL

1. Data Definition

`createdb` *dbname*	*same*
`destroydb` *dbname*	*same*
`create` *relname* (*attr1*=*type1* { , *attri*=*typei*}) `inherits(`*relname1* { , *relnamei* ...}) `archive =` *archive_mode* `store = "`*smgr_name*`"` `arch_store = "`*smgr_name*`"`	`CREATE TABLE` *relname* (*attr1 type1* { , *attri typei* ...}) `INHERITS(`*relname1* { , *relnamei* ...}) `ARCHIVE =` *archive_mode* `STORE = "`*smgr_name*`"` `ARCH_STORE = "`*smgr_name*`";`
`destroy` *relname1* { , *relnamei* ...}	`DROP TABLE` *relname1* { , *relnamei* ...};
`create` [`forward`|`backward`] `version` *relname* `from` *relname2* [`[ abstime ]`]	`CREATE` [`forward`|`backward`] `VERSION` *relname* `FROM` *relname2* [`[ abstime ]`];
`addattr` (*attr1*=*type1* { , *attri* = *typei* ...}) `to` *relname* [*]	`ALTER TABLE` *relname* [*] `ADD COLUMN` *attr1 type1*;
`rename` *attr1* `in` *relname* [`*`] `to` *attr2*	`ALTER TABLE` *relname* [`*`] `RENAME` [`COLUMN`] *attr1* `TO` *attr2*;
`rename` *relname1* `to` *relname2*	`ALTER TABLE` *relname1* `RENAME TO` *relname2*;
`change acl` [`group`|`user`] *name*[`+-=`][`arwR`] relname1 { , *relnamei* ...}	*same*
`define view` *view_name* ([*dom_name1* =] *expr1* {, [*dom_namei* =] *expri* ... }) [`from` *from_list*] [`where` *qual*]	`CREATE VIEW` *view_name* `AS` `SELECT` *expr1* [`AS` *attr1*] { , *expri* [`AS` *attri*] ...} [`FROM` *from_list*] [`WHERE` *qual*];
`remove view` *viewname*	`DROP VIEW` *viewname*;
`define` [`archive`] `index` *indexname* `on` *relname* `using` *access_method* ({*attr1* | *funcname*(*attr1* { , *attri* ...})} *type_class1*) [`where` *qual*]	`CREATE INDEX` *indexname* `ON` *relname* `USING` *access_method* ({*attr1* | *funcname*(*attr1* { , *attri* ...})} *type_class1*) [`WHERE` *qual*];

```
extend index indexname [where qual]
```

same

```
remove index indexname
```

```
DROP INDEX indexname;
```

2. User Defined Types, Functions, etc.

```
define type typename (
    internallength = (number | variable),
    [ externallength = (number | variable), ]
    input = input_function,
    output = output_function
    [, element = typename]
    [, delimiter = character]
    [, default = "string" ]
    [, send = send_function ]
    [, receive = receive_function ]
    [, passedbyvalue]
    )
```

```
CREATE TYPE typename (
    internallength = (number | variable),
    [ externallength = (number | variable), ]
    input = input_function,
    output = output_function
    [, alignment = { double | int } ]
    [, element = typename]
    [, delimiter = character]
    [, default = "string" ]
    [, send = send_function ]
    [, receive = receive_function ]
    [, passedbyvalue]
    );
```

```
remove type typename
```

```
DROP TYPE typename;
```

```
define function function_name (
    language = {"c" | "postquel"},
    returntype = type-r
    [ , iscachable ]
    [ , trusted = {"t" | "f"} ]
    [ , percall_cpu = costly{!*} ]
    [ , perbyte_cpu = costly{!*} ]
    [ , outin_ratio = percentage ]
    [ , byte_pct = percentage ]
    )
    arg is ( [ type1 { , typei ... } ] )
    as {"objectfilename" | "postquel-queries"}
```

```
CREATE FUNCTION function_name
    ( [ type1 { , typei ... } ] )
    RETURNS type-r
    [WITH ([ iscachable ]
    [ , trusted = {"t" | "f"} ]
    [ , percall_cpu = costly{!*} ]
    [ , perbyte_cpu = costly{!*} ]
    [ , outin_ratio = percentage ]
    [ , byte_pct = percentage ]
    )]
    AS {"objectfilename" | "sql-queries"}
    LANGUAGE {"c" | "sql"};
```

```
remove function funcname ([type1 { , typei ...}])
```

```
DROP FUNCTION funcname ([type1 { , typei ...}]);
```

```
define operator operator_name (
    [ arg1 = type-1 ]
    [ , arg2 = type-2 ]
    , procedure = func_name
    [, precedence = number ]
    [, associativity = {left | right | none | any}
]
    [, commutator = com_op ]
    [, negator = neg_op ]
    [, restrict = res_proc ]
    [, hashes]
    [, join = join_proc ]
    [, sort = sor_op1 { , sor_opi ... } ]
    )
```

```
CREATE OPERATOR operator_name (
    [ leftarg = type-1 ]
    [ , rightarg = type-2 ]
    , procedure = func_name
    [, precedence = number ]
    [, associativity = {left | right | none | any}
]
    [, commutator = com_op ]
    [, negator = neg_op ]
    [, restrict = res_proc ]
    [, hashes]
    [, join = join_proc ]
    [, sort = sor_op1 { , sor_opi ... } ]
    );
```

```
remove operator opname({ltype, rtype | type})
```

```
DROP OPERATOR opname({ltype | none},
    {rtype | none});
```

```
define aggregate agg-name [as] (
    [sfunc1 = state-transition-function-1,
    basetype = data-type,
    stype1 = sfunc1-return-type]
    [, sfunc2 = state-transition-function-2,
    stype2 = sfunc2-return-type]
    [, finalfunc = final-function]
    [, initcond1 = initial-condition-1]
    [, initcond2 = initial-condition-2]
    )
```

```
CREATE AGGREGATE agg-name [AS] (
    [sfunc1 = state-transition-function-1,
    basetype = data-type,
    stype1 = sfunc1-return-type]
    [, sfunc2 = state-transition-function-2,
    stype2 = sfunc2-return-type]
    [, finalfunc = final-function]
    [, initcond1 = initial-condition-1]
    [, initcond2 = initial-condition-2]
    );
```

```
remove aggregate aggname
```

```
DROP AGGREGATE aggname;
```

```
define [instance | rewrite] rule rule_name
    is on { retrieve | replace | delete | append }
    to object [where qual]
    do [instead]
    [action | nothing | [actions ...]]
```

```
CREATE RULE rule_name AS
    on { select | update | delete | insert }
    TO object [WHERE qual]
    DO [instead] [action | nothing | [actions ...]];
```

```
remove [rewrite] rule rulename
```

```
DROP RULE rulename;
```

3. Data Manipulation

```
append relname
    (attr1=expr1 {, attri=expri ...})
    [from from_list]
    [where qual]
```

```
INSERT INTO relname [ (attr1 {, attri ...}) ]
    { VALUES (expr1 {, expri ...}); |
    SELECT expr1 {, expri ...}
        [FROM from_clause]
        [WHERE qual] ; }
```

```
delete instance_var
    [from from_list]
    [where qual]
```

```
DELETE FROM relname
    [WHERE qual];
```

```
replace instance_var
    (attr1=expr1 {, attri=expri ...})
    [from from_list]
    [where qual]
```

```
UPDATE relname
    SET attr1=expr1 {, attri=expri ...}
    [WHERE qual];
```

```
retrieve [into relname] [unique]
    (attr1=expr1 {, attri=expri ...})
    [from from_list]
    [where qual]
    [sort by attr1 [using op1] {, attri [using opi] ...
}]
```

```
SELECT [distinct]
    expr1 [AS attr1] {, expri [AS attri] ...}
    [INTO TABLE relname]
    [FROM from_list]
    [WHERE qual]
    [GROUP BY attr1 {, attri ...}]
    [ORDER BY attr1 [USING op1] {, attri [USING opi]
... }];
```

`retrieve portal` *portalname* [`unique`] `(`*attr1*`=`*expr1* {`,` *attri*`=`*expri* ...}) [`from` *from_list*] [`where` *qual*] [`sort by` *attr1* [`using` *op1*] {`,` *attri* [`using` *opi*]}]	`DECLARE` *portalname* [`BINARY`] `CURSOR FOR` `SELECT` [`distinct`] *expr1* [`AS` *attr1*] {`,` *expri* [`AS` *attri*] ...} [`FROM` *from_list*] [`WHERE` *qual*] [`ORDER BY` *attr1* [`using` *op1*] `,` *attri* [`using` *opi*]];
`fetch` [`forward` | `backward`] [*number* | `all`] [`in` *portalname*]	*same*
`close` *portalname*	`CLOSE` *cursorname*;

4. Transactions

`begin` [`transaction`]	*same*
`abort` [`transaction`]	*same*
`end` [`transaction`]	*same*

5. Miscellaneous

`copy` [`binary`] [`nonulls`] *relname* {`to`|`from`} {*filename*|`stdin`|`stdout`} [`using` *maprelname*]	*same*
`load` *filename*	*same*
`purge` *relname* [`before` *abstime*] [`after` *reltime*]	*same*
`vacuum`	`VACUUM` [*relname*]
none	`CLUSTER` *indexname* `ON` *relname*

notes:

1. keywords are case-insensitive. WHERE is the same as wHeRe
2. comments starts with `--`
3. some postquel queries are inexpressible in SQL (eg. inheritence queries in updates and deletes)